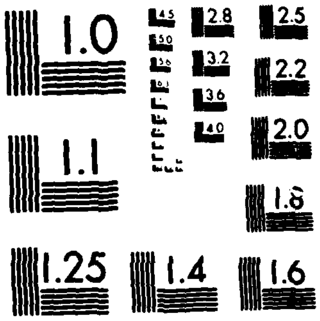
AD-A164 034 PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT 1/5
MANEUVERING TARGET..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R B ANDERSON
UNCLASSIFIED DEC 85 AFIT/GE/ENG/85D-2 F/G 19/5 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

1

PRELIMINARY KALMAN FILTER DESIGN
TO IMPROVE AIR COMBAT MANEUVERING
TARGET ESTIMATION FOR THE
F-4E/G FIRE CONTROL SYSTEM

THESIS

Ross B. Anderson, BSEEE
Captain, USAF

AFIT/GE/ENG/85D-2

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/85

1

PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT MANEUVERING TARGET ESTIMATION FOR THE F-4E/G FIRE CONTROL SYSTEM

THESIS

Ross B. Anderson, BSEEE
Captain, USAF

AFIT/GE/ENG/85D-2

AFIT/GE/ENG/85D-2

PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT MANEUVERING TARGET ESTIMATION FOR THE F-4E/G FIRE CONTROL SYSTEM

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Masters of Science in Electrical Engineering

Ross B. Anderson, BSEEE
Captain, USAF

December 1985

Approved for public release; distribution unlimited

Preface

Currently, the F-4E/G uses a Wiener-Hopf filter for estimating target position, velocity, and acceleration during air combat maneuvering. As implemented, the estimates contain unacceptable errors. The purpose of this study is to determine the feasibility of replacing the Wiener-Hopf filter with a Kalman filter in order to obtain better estimates. The determination is made by first designing an appropriate Kalman filter and then testing the design through computer simulation.

In performing the research and writing for this thesis I have had the help and support of others. I am deeply indebted for the professional help I received. First I would like to thank the AFIT library and computer room staffs for services well rendered. Second, I would like to thank Mr. Stan Musick and Mr. Ralph Bryan of the Avionics laboratory and Captain Bert Halbert of AFIT for their support and insights. Third, I would like to thank Mr. Bob Beal and Mr. Chuck Strickler of OO-ALC/MMECB for their continual support and insights. Fourth, I would like to thank my thesis committee members, Major William H. Worsley and Professor Peter S. Maybeck. Finally, my greatest appreciation goes to my family for their understanding. I only hope I can be as understanding of their problems as they were of mine when I was working on the thesis.

Ross B. Anderson

Contents

Page

List of Figures

List of Tables

List of Symbols and Abbreviations

Page where first defined or used

a_{fx}, a_{fy}, a_{fz}	35	A_Z	17
A_{ZE}	25	A_{ZT}	40
B	33	$\underline{B}_d$	57
C	28	c.g.	34
$E[\cdot]$	48	EKF	45
E_L	17	E_{LE}	25
E_{LT}	40	$\underline{F}$	33
$\underline{F}[t;\underline{x}(t/t_i)]$	50	$\underline{f}[\underline{x}(t),\underline{u}(t),t)]$	47
$\underline{G}$	33	$\underline{G}(t)$	47,48
H_{11} through H_{69}	51	$\underline{H}[t_i;\underline{x}(t_i^-)]$	49
$\underline{h}[\underline{x}(t_i),t_i]$	12	i,j,k	25
i_1, j_1, k_1	25	i_o, j_o, k_o	13
$\underline{K}(t_i)$	49	l	8
l,m,n	19	LSB	15
N,E,D	19	$\underline{P}(t_i)$	49
$\underline{P}(t_i^+)$	48	$\underline{P}(t_i^-)$	48
$\underline{P}_o$	47,56	$\underline{Q}$	46,48
$\underline{Q}_d$	58	Q_1 through Q_3	54,55
$\underline{Q}_{11}$ through Q_{99}	59	R	17
$\underline{R}$	42	RDOT or R	17
$\underline{R}_{nom}$	16	SAS	78
SSDC	85	t_i, t_i^-, t_{i-1}^+	78
$\underline{u}(t)$	48	$v_{fx,y,z}$	31
$\underline{v}(t_i)$	12	w_j	17
w_k	17	$w_{x,y,z}$	31

$\hat{\underline{x}}(t_i^{+})$	48	$\hat{\underline{x}}(t_i^{-})$	48
x,y,z	19	$\hat{\underline{x}}_o$	47,55
x_1 through x_9	28,29	$\underline{z}(t_i)$	12
Z_1 through Z_6	37-41		54
δ	48	τ or tau	48
$\underline{\Phi}$	33	Ψ	19
θ	19	ϕ	19
Δt	33	$\mathcal{L}^{-1}$	33

Abstract

Currently, the F-4E/G uses a Wiener-Hopf filter for estimating target position, velocity, and acceleration during air combat maneuvering. As implemented, the errors between the actual target variables and the estimate of these variables are too large. The purpose of this study is to evaluate the feasibility of replacing the Wiener-Hopf filter with a Kalman filter in order to obtain better estimates. The evaluation is made by first designing an appropriate preliminary design Kalman filter and then testing the design through a Monte Carlo computer simulation analysis. The computer simulation results indicate that the Kalman filter is capable of significantly outperforming the Wiener-Hopf filter, and as such, should be developed into a final design.

The Kalman filter contains nine states (three relative target position, three total target velocity, and three total target acceleration states). Filter propagation is based on linear time-invariant dynamics primarily because of the limited capabilities of the on-board aircraft computer. The linear dynamics permits propagation by a state transition matrix. Measurement updates use six measurements (range, range rate, azimuth angle, elevation angle, azimuth rate, and elevation rate) available on the F-4. Both continuous time sampled-data and discrete-time sampled-data designs are included.

PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT MANEUVERING TARGET ESTIMATION FOR THE F-4E/G FIRE CONTROL SYSTEM

I. INTRODUCTION

1.1 Motivation: Organizational Problem

There are more than 3500 F-4 aircraft in service worldwide today, and the Pentagon predicts that over 2000 will still be in service in the year 2000 (1:16-18). This prediction has resulted in the realization that the F-4 will remain in the USAF inventory much longer than initially planned (1:16-18). As a consequence, the Air Logistics Center (ALC) at Hill AFB is continually upgrading F-4E/G aircraft systems to improve aircraft survivability and effectiveness in hostile zones of operation. Improved capability directly influences the future utilization of F-4E/G aircraft. In order to support the upgrade, OO-ALC/MMECB has requested Air Force Institute of Technology (AFIT) assistance in improving F-4E/G air combat maneuvering algorithms for target estimation (2:1-2). An improved air combat maneuvering capability will increase the effectiveness of this aircraft in the air-to-air role and should increase the air combat survivability of the fighter.

1.2 Present Situation

In an air-to-air role an aircraft must first be able to maneuver to elude the enemies' fire power and second be capable of delivering its own fire power. The overall objective is to be able to launch a missile against a target when the probability of kill is high (most USAF F-4s do not have guns and they rely primarily on air-to-air missiles for fire power). However, when F-4E/G aircraft maneuver during an air-to-air engagement, the fire control system which predicts when the target is in the envelope of vulnerability becomes unstable. As currently implemented on the F-4E/G, large, unacceptable errors in target position, velocity, and acceleration estimates result. Pilots who fly the F-4E/G models have complained that the target steering dot oscillates and is distracting, especially during high roll rate maneuvers (3:1). As a partial solution, a post filter has been added to detect when the plane starts to roll and then "fixes" the steering dot in place. This results in the steering dot sometimes "jumping around" on the pilot's display after the dot is released from the "fixed" position. Thus, the air combat maneuvering target information provided to the pilot is of questionable value and may result in launching a missile outside its effective envelope and consequently missing its target. As such, when the pilot needs target information most, the steering dot may either be oscillating, fixed in place, or not used because of pilot distrust.

OO-ALC/MMECB engineers attribute the unstable nature of the steering dot to "noisy and inaccurate line-of-sight rates" and the use of a Wiener-Hopf filter for providing target estimates (2:1-2). It may be possible to stabilize the steering dot by either replacing the radar system with an improved system, or by replacing the Wiener-Hopf filter with a Kalman filter, or both. The USAF is not willing to request or provide funds to update the radar system (1:16-18). Thus at the present time, the only viable solution is to develop and test a dynamic software filter in order to try to eliminate or significantly reduce the distracting steering dot oscillation and jump. This solution is a low cost option because it does not involve hardware changes.

Better estimates may theoretically be obtainable by developing adequate state dynamic models, measurement models, time propagation models, and measurement update equations and then incorporating these into a properly tuned Kalman filter. However, an actual history of radar system noises and measurement noises is not accurately known. Without actual noise data, it is not possible to tune a Kalman filter for implementation. The tuned filter in this thesis is based on values of actual noises from one flight test (nonmaneuvering aircraft and target) and truth model dynamic radar lags. As such, a sensitivity analysis based on simulation techniques for various types of noises is desirable for follow on testing in OO-ALC test facilities and test aircraft.

1.3 F-4E/G System Limitations

The F-4E/G fire control system is not a state-of-the-art system, which places constraints on the design of an Kalman filter. The limitations are:

1. The fire control computer, the LRU-1, is a fixed point, 16 bit wordlength machine.
2. The analog to digital convertors provide only 10 significant bits.
3. The LRU-1 operates at 300,000 operations per second.
4. The total memory in the LRU-1 which can be allocated for the target estimation is 8K words which must be shared with a long range intercept (LRI) algorithm (concurrently being modified in another thesis (4)).
5. Target estimation update is currently required every 40 msec. An update period range between 40 and 100 msec is required.

The fixed point and analog to digital conversion restrictions, as well as other restrictions, impact on the overall design philosophy. The operating time and memory restrictions imply the need for efficient yet accurate reduced order model Kalman filters. The memory restrictions led to an early realization that air combat maneuvering and long range intercept (a parallel Air Force Institute of Technology Thesis (4)) algorithms must be shared to make efficient use of available memory. For the design to be useful in the real world, simulation testing must account for the above restrictions.

1.4 Study Objective

The primary emphasis of this study is to design and test Kalman filter algorithms for estimating target position, velocity, and acceleration to determine the feasibility of replacing the current Wiener-Hopf filter with a Kalman filter for F-4E/G air combat maneuvering. Air combat maneuvering, for this thesis, is defined as air-to-air combat below 32,000 feet of elevation and within an 8 nautical mile radius of the F-4 aircraft. The intent is first to provide sufficient theory on which a number of Kalman filters can be designed for this particular problem. Then, to propose a nine state reduced order Kalman filter as a preliminary design considering the system restrictions described in Section 1.3. Next, the preliminary design Kalman filter is tested through computer simulation to validate its performance. Finally, recommendations are made on areas where further research may be warranted. Every attempt is made to model or account for the system restrictions so the final product will be of use on real world F-4E/G aircraft.

1.5 Scope and Organization

This study follows a systematic design procedure which parallels a procedure proposed by Maybeck (5:341-342). Due to the limited time available for this effort, it is not possible to complete in full detail all of the steps of the systematic design process. The procedure is included to illustrate the design approach employed. The procedure is as follows (with responsibilities defined).

1. Development of a "truth model" based on the system dynamics and measurement models (this study, Chapters II, III, and IV and trajectory simulation program from OO-ALC).

2. Development of the Kalman filter theory based upon the "truth model" (this study, Chapter III).

3. Proposal of a simplified, reduced order Kalman filter based on system models and F-4 E/G system limitations (this study, Chapter III).

4. Development of test trajectory algorithms (this study, Chapter IV).

5. Completion of a Monte Carlo analysis (sensitivity analysis) on the selected reduced order Kalman filter proposed (this study, Chapter V).

6. Completion of a thorough Monte Carlo analysis based on designs showing the most promise (either future study or OO-ALC/MMECB).

7. Completion of a performance / computer loading tradeoff analysis and selection of a design (partially this study, Chapter IV, Stand-Alone Simulation (SAS) Program, and Appendix F: future research or efforts by OO-ALC warranted). The design included in the SAS will be proposed to OO-ALC for possible implementation.

8. Implementation of the chosen designs on the online computer used in the F-4E/G (OO-ALC).

9. Completion of checkout, final tuning, and

operational tests of the filter (OO-ALC).

1.6 Literature Review

As a service to the reader and OO-ALC, the contents or abstracts and important aspects of many references pertaining to this study are listed in Appendix A.

II. ANALYTICAL DEVELOPMENT (MODELS)

2.1 Introduction

This chapter contains the necessary concepts, geometry, and system equations for mathematically modeling an air-to-air engagement for implementation on a digital computer. The models in this chapter are specific for the F-4E/G but can easily be adapted to a wide class of problems. As an example of the specific design, the radar model is for a space stabilized gimballed radar; strapdown models are not addressed, but can be found in other references (6:23-32). The overall intent is to provide the necessary background, system restrictions, assumptions, and data required so that a reader with air-to-air scenario background can logically follow the development. To accomplish this, the chapter is divided into the following subsections: coordinate system, radar description, dynamics model, simulated measurement model (radar), and the truth model.

2.2 Coordinate System, Basic Concepts

The air-to-air engagement can be modeled in either a Cartesian coordinate frame or a spherical coordinate frame. The Cartesian coordinate frame is referenced to a flat earth approximation and the spherical coordinate frame is referenced to the line-of-sight (1) or the tracker frame. The choice is arbitrary, but impacts on both the measurement model (radar) and the system dynamics model. A Cartesian coordinate system is used throughout this thesis for the reasons discussed below.

Using a Cartesian coordinate frame (located at the aircraft center of gravity (c.g.), flat earth app:oximation) results in a linear model for target dynamics. The angular rate terms which appear in differential equations written in a rotating frame are eliminated. In other words, the Coriolis and centripetal acceleration terms are eliminated from the system model. A slight variation is a Cartesian reference frame that rotates about the c.g. (i.e., moving the origin of the frame from aircraft center of gravity to radar tracker frame). The distance from the c.g. to the tracker frame and rotation rates in velocity and acceleration estimates can be accounted for by applying dynamic equations and relations. Linear target dynamics are retained. Radar units are typically in the nose of the fighter aircraft (not the center of gravity), thus the radar is often modeled in a Cartesian frame located a fixed distance from the c.g. This is the frame used throughout this study. Figure 2-1 illustrates the antenna tracker frame used.

When linear target dynamics are preserved, it is possible to form a state transition matrix resulting in an easier and more computational efficient digital implementation. The price one must pay for this simpler dynamic system is a nonlinear measurement (radar) model.

A spherical reference frame, on the other hand, results in linear measurement models but nonlinear target dynamics equations. For the air-to-air problem, the nonlinear dynamic

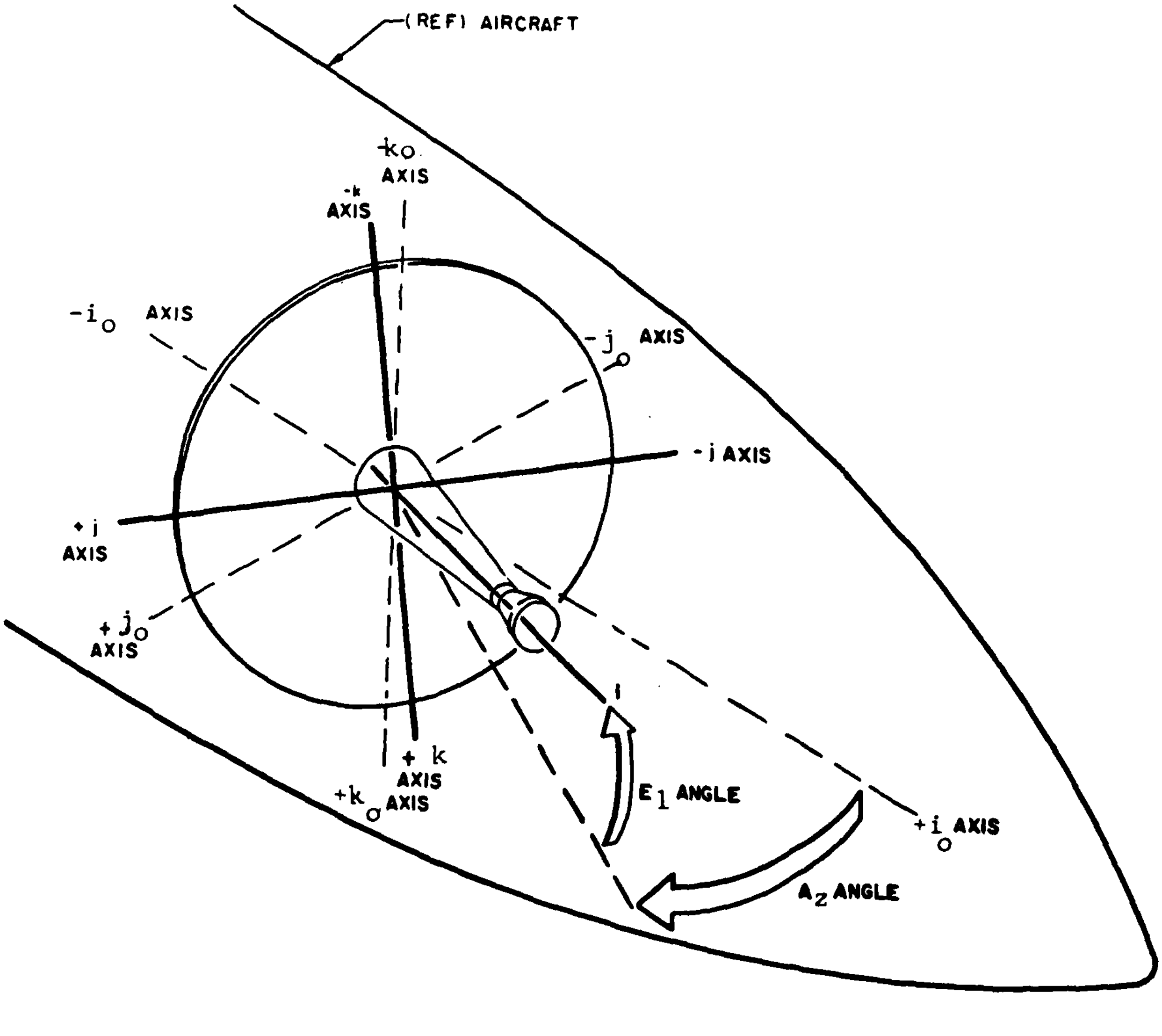


Figure 2-1. Antenna Tracker Reference (i_o,j_o,k_o) and Antenna Coordinates (i,j,k)
(Adapted from T.O. 12P2-2APQ120-2-3-7)

equations result in nonlinear propagation equations which require on-line integration. Conversely, the linear, time-invariant dynamic equations associated with the Cartesian coordinate frame result in algorithms that do not require on-line integration to compute the state transition matrix (7:1-28). Therefore, the target dynamics are usually modeled with respect to a Cartesian coordinate frame, while the radar measurements typically consist of at least vehicle slant range, azimuth, and elevation (7).

Without loss of generality, a Cartesian coordinate system is used in following subsections on development of radar models and system dynamic models. This is consistent with the current F-4E/G filter design, since the P004 update documentation states, "critical measurements of target position and velocity relative to the F-4E/G are made in radar antenna coordinates, it would be very convenient to express the filter inputs...in this same triad" (8:3-423). As such, the primary reference frame is the antenna tracker reference frame.

2.3 Radar Description

2.3.1 Introduction

The study of radar can be a vast and complicated process; a thesis all in itself. OO-ALC/MMECB has developed a radar simulation model which is modified and used in this thesis (see Chapter IV and Appendix B). The intention of this section is to develop and discuss only the necessary radar concepts and information to understand the quality and quantity of the radar inputs to the F-4E/G fire control system. The characteristics of the measurement noise, $\underline{v}(t_i)$ are established for the radar measurement equation (see Section 3.2.1 for further explanation) as

$$\underline{z}(t_i) = \underline{h}[\underline{x}(t_i),t_i] + \underline{v}(t_i) \qquad (2\text{-}1)$$

where

$\underline{z}(t_i)$ = noise corrupted vector measurements at time t_i,

$\underline{h}[\underline{x}(t_i),t_i]$ = nonlinear measurement function, and

$\underline{v}(t_i)$=discrete zero mean white Gaussian noise at time with covariance $\underline{R}(t_i)$.

2.3.2 Radar System, Basic Concepts

In simplistic terms, the F-4E/G APQ-120 radar is a system which transmits an electromagnetic waveform and receives back a reflected portion of the waveform. From this reflected waveform, the system provides measurements of range, range rate, azimuth, elevation, azimuth rate, and elevation rate. The measurements are relative, from the apparent radar centroid of the target to the attacker F-4 radar. The information can be represented in either Cartesian or line-of-sight reference frames, but, as discussed in Section 2.2, this study utilizes the Cartesian frame. Additionally, the F-4E/G radar is a non-Doppler, gimballed, air intercept (AI), space stabilized system. The radar antenna rotates about a fixed point in the aircraft relative to a radar reference frame i_o, j_o, and k_o (see Figure 2-1). Rate gyros and resolvers on the radar antenna provide azimuth and elevation rates and azimuth and elevation angles, respectively, to the radar reference frame. Functionally, once a target is detected in a search mode, the aircrew or the radar system itself can lock on and track the target, providing the measurements listed above. Then the measurements are used by the fire control system to compute when a target is in range for firing a missile against the target. In a practical sense, one major problem remains: determining how good or how accurate the measurements really are.

2.3.3 Radar System, Measurement Corruption

Radar measurement noises degrade the quality of information provided to the fire control system. Since the measurements are used in estimating target position, velocity, and acceleration, the noises must be addressed. The noise sources are briefly discussed; the intent is to only provide representative noise values for the available F-4E/G radar measurements: range, range rate, azimuth, elevation, azimuth rate and elevation rate. According to Barton and Ward (9:Chapter 8), radar noise can arise from a large number of sources. Table II-1 lists common noise sources (refer to Barton and Ward for definition of the noise sources).

Table II-1

Common Radar Noise Sources

	Angle Error	Range Error
Thermal Noise	X	X
Clutter and Interference	X	X
Multipath Reflections	X	X
Target Glint and Scintillation	X	X
Quantization and Array Error	X	X
Dynamic Lag	X	X
Atmosphere Propagation	X	X
Monopulse Network Error	X	
Servo and Mechanical Error	X	
Receiver Time Delay Instability		X
Time Discriminator Alignment and Stability		X
Servo Loop Noise		X
Error in Converting Measured Delay to output data (apart from quantizing noise)		X
Reference Oscillator Frequency Stability		X

Quite often, radar noises are modeled as being dominated by just the noise from glint and scintillation (scintillation is commonly called amplitude distribution (9:171)). For example, the last update for the OFP ACM Computer Modification (8:3-413) states "the major sources of measurement noise in target velocity are radar generated glint and amplitude noise sensed by the antenna mounted rate gyros." However, calculations from this study indicate that additional noise sources for the F-4E/G radar system are introduced through antenna dynamics and the analog to digital (A/D) conversion (quantization in Table II-1). Antenna dynamics are discussed in Section 2.5.2. The A/D conversion uses only 10 significant bits. This results in the least significant bit (LSB) adding noise to the measurement. The added noise can be calculated as follows

$$\text{LSB errors} = \frac{\text{Maximum Measurement Interval Size}}{2^{10}} \qquad (2\text{-}2)$$

Modeling the LSB error as an uniform random variable, mean and standard deviation formulas for both a truncated and rounded A/D conversion process are provided in Table II-2 (see (5:92-93) for a derivation). Using these relations,

Table II-2

Mean and Standard Deviation Equations for A/D Process

	mean	standard deviation
Truncated Case	LSB/2	$LSB/(12)^{1/2}$
Rounded Case	0	$LSB/(12)^{1/2}$

LSB errors are illustrated in Table II-3 (using maximum measurement values from F-4 Improved Air-to-Air Missile Program (10:4-3)). A Gaussian approximatiom (using the mean and standard deviation) of the noise process introduced by the A/D process is used in forming $\underline{R}_{nom}$ in Equation (2-3).

Table II-3

Mean and Standard Deviation for A/D Process

	LSB Error	mean[1]	mean[2]	S.D.
Range(<60,000 feet)	58.59	29.29	0	16.91
Range Rate(ft/sec)	1.93	0.97	0	0.56
Azimuth(radians)	0.00127	0.00064	0	0.000366
(degrees)	0.0728	0.0364	0	0.0210
Elevation(radians)	0.00127	0.00064	0	0.000366
(degrees)	0.0728	0.0364	0	0.0210
Azimuth Rate(rad/sec)	0.000511	0.00256	0	0.00148
(deg/sec)	0.0293	0.01465	0	0.00846
Elevation Rate(rad/sec)	0.000511	0.00256	0	0.00148
(deg/sec)	0.0293	0.01465	0	0.00846

1- mean for the truncated case
2- mean for the rounded case

Without a sufficient history of radar noise data, nominal values are selected based on data from OO-ALC/MMECB

(11), previous studies, and the above A/D noise data. The nominal values are presented in Table II-4 and in matrix form in Equation (2-3).

Table II-4

Nominal Noise Values for the $\underline{R}$ Matrix

Measurement	Symbol	Standard Deviation
Range	R	17.0 feet
Range Rate	RDOT	16.0 ft/sec
Azimuth Angle	A_Z	2.27 mrad
Elevation Angle	E_L	2.27 mrad
Azimuth Rate	AZDOT or w_k	12.22 mrad/sec
Elevation Rate	ELDOT or w_j	12.22 mrad/sec

$$\underline{R}_{nom} = \begin{bmatrix} 289 & 0 & 0 & 0 & 0 & 0 \\ 0 & 256 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5.15\text{x}10^{-6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 5.15\text{x}10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.49\text{x}10^{-4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.49\text{x}10^{-4} \end{bmatrix} \quad (2\text{-}3)$$

2.3.4 Selected Radar Functions

The development to this point considers a space stabilized radar that is locked-on and tracking a target providing noise corrupted measurements (minus antenna dynamics which are added in Section 2.5.2) which are representative of the F-4 radar. A radar simulation model obtained from OO-ALC/MMECB (12) includes the F-4E/G APQ-120 radar servo dynamics and provides outputs of measurements of range, range rate, antenna azimuth, antenna elevation, azimuth rate, and elevation rate. The outputs are used as inputs to a Kalman filter. The radar servo model includes radar antenna dynamics and closed-loop control to provide tracking. The only assumptions made are: 1) at the start of a simulation run, the radar is already locked onto the target, and 2) the radar tracks the target (within gimbal limits of plus or minus 60 degrees) during the simulation.

Using the OO-ALC radar model, with noise strengths from Table II-4, it is now possible to formulate a measurement model. Since this model is based on the target estimation state vector, it is necessary to develop the target estimation filter geometry and dynamics model.

2.4 System Dynamics Model (Radar Reference State Equations)

2.4.1 Introduction

A system dynamics model is required to obtain position, velocity, and acceleration estimates of target. In previous sections, it is stated the radar is space stabilized. Noise corrupted measurement of range, range rate, azimuth angle, elevation angle, azimuth rate, and elevation rate are available. As discussed in Section 2.2, it is desired to develop models in a Cartesian frame resulting in linear dynamic equations. For the tracking problem, it is possible to develop a linear time-invariant dynamics model which allows efficient implementation through a state transition matrix.

2.4.2. Geometry, Coordinates, and Transformations

Using a Cartesian frame, it is now possible to describe the problem geometry, assign a coordinate system, and develop coordinate transformations from one frame to another. Descriptive illustrations (Figures 2-2 through 2-6) are used to facilitate the discussion.

The geometry and notation is consistent with the F-4E/G P004 update (8:Chapter 3). Euler rotations of yaw (Ψ), pitch (θ), and roll (ϕ) are employed. Figure 2-2 illustrates aircraft (l,m,n), space (x,y,z) and geographic (N,E,D) frames with appropriate Euler angles. Note that the geographic and space frames differ by only a rotation in heading. Further, note that the aircraft frame is obtained by successive rotations of pitch and roll from the space frame. Figure 2-3 also ilustrates this process.

To be consistent with the F-4E/G aircraft, a two degree offset in pitch is included between the aircraft (l,m,n) and radar reference (i_o,j_o,k_o) frames. Figure 2-4 illustrates this.

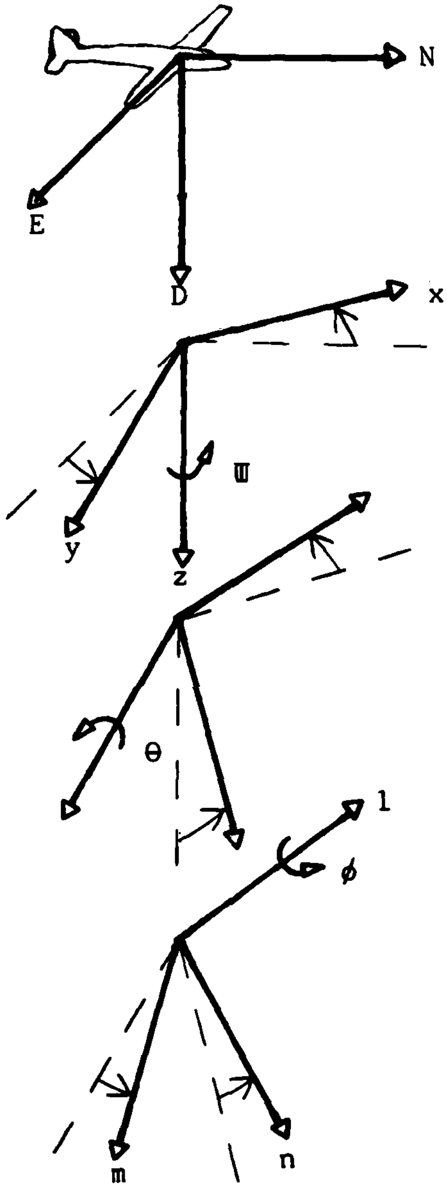
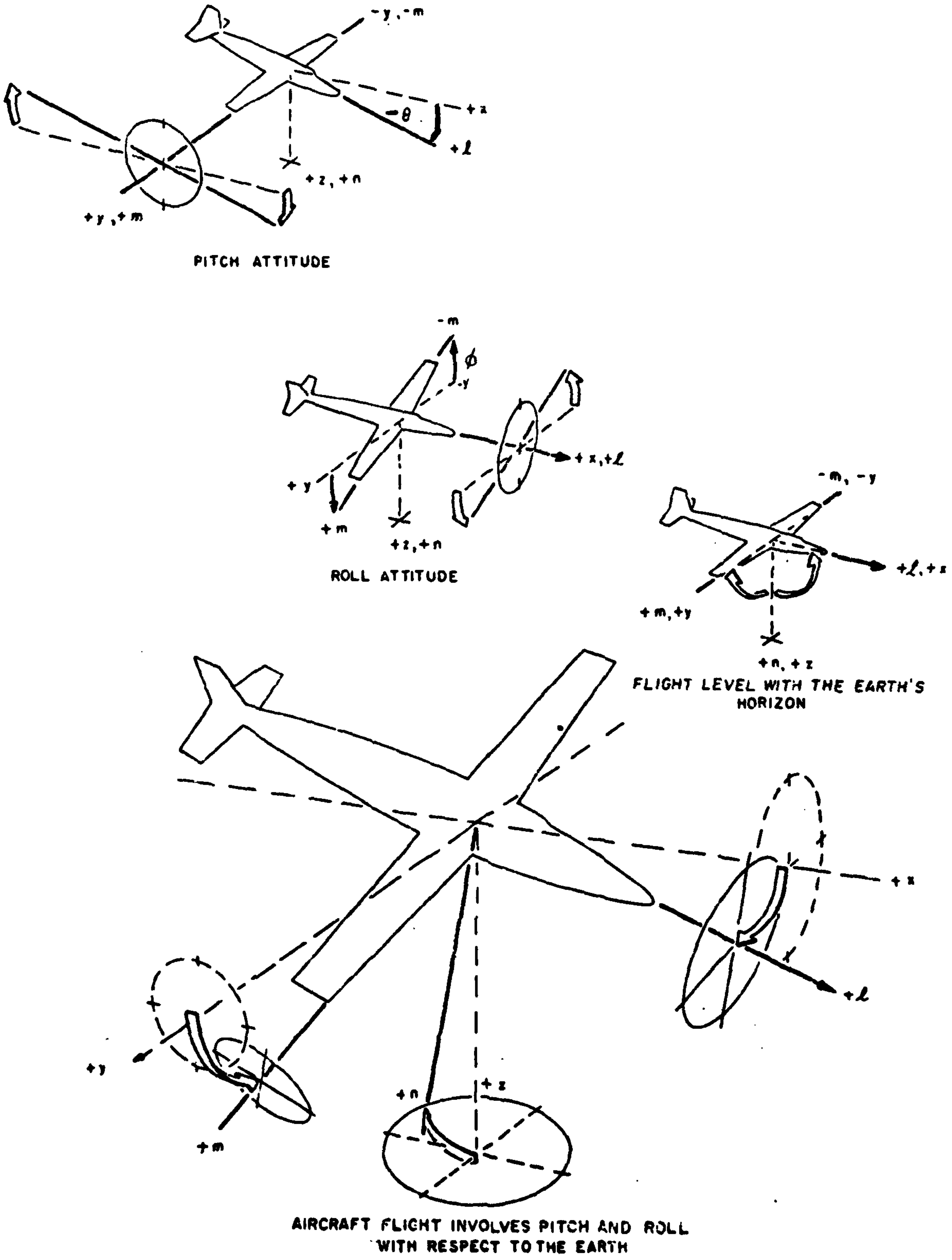


Figure 2-2 Geographic Axes (N,E,D), Space Axes (x,y,z), Aircraft Axes (l,m,n), and Euler Rotations



PITCH ATTITUDE

ROLL ATTITUDE

FLIGHT LEVEL WITH THE EARTH'S HORIZON

AIRCRAFT FLIGHT INVOLVES PITCH AND ROLL WITH RESPECT TO THE EARTH

Figure 2-3 Space and Aircraft Coordinates (Adapted from T.O. 12P2-2APQ120-2-3-7)

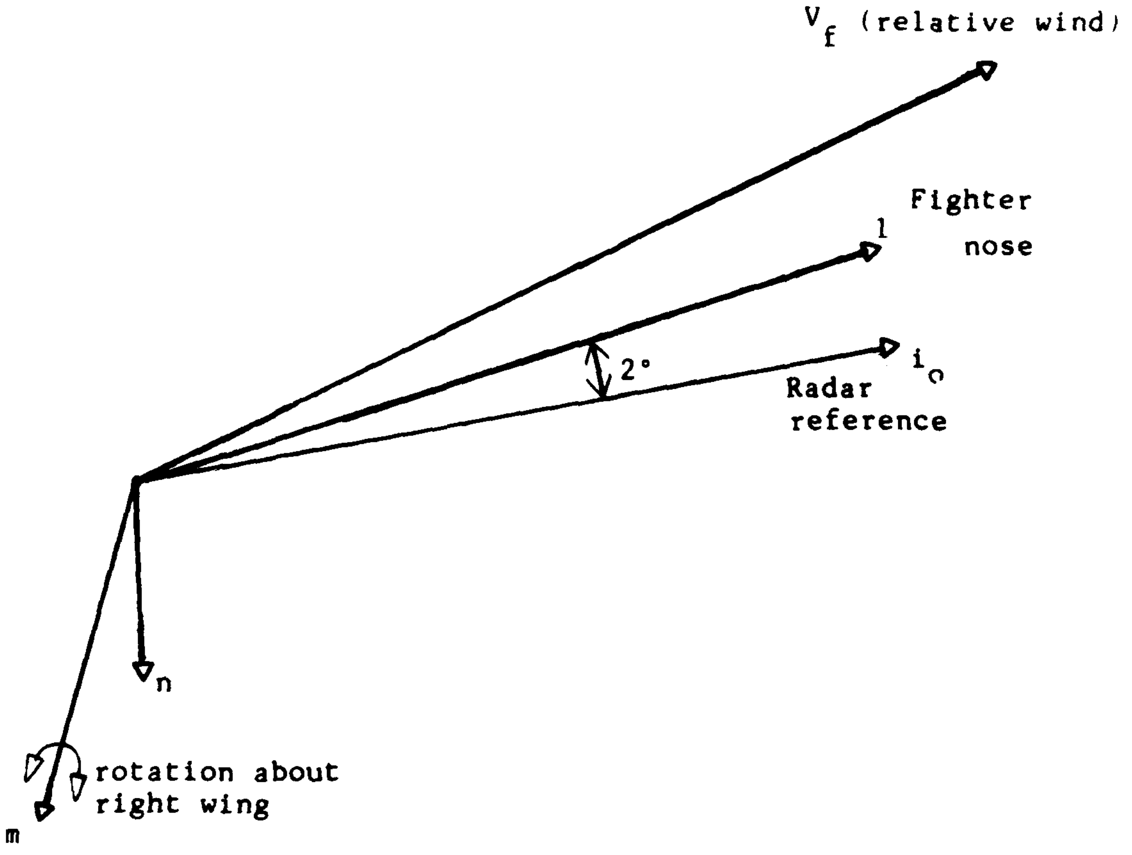


Figure 2-4 Relative Wind, Aircraft, and Radar Reference Axes

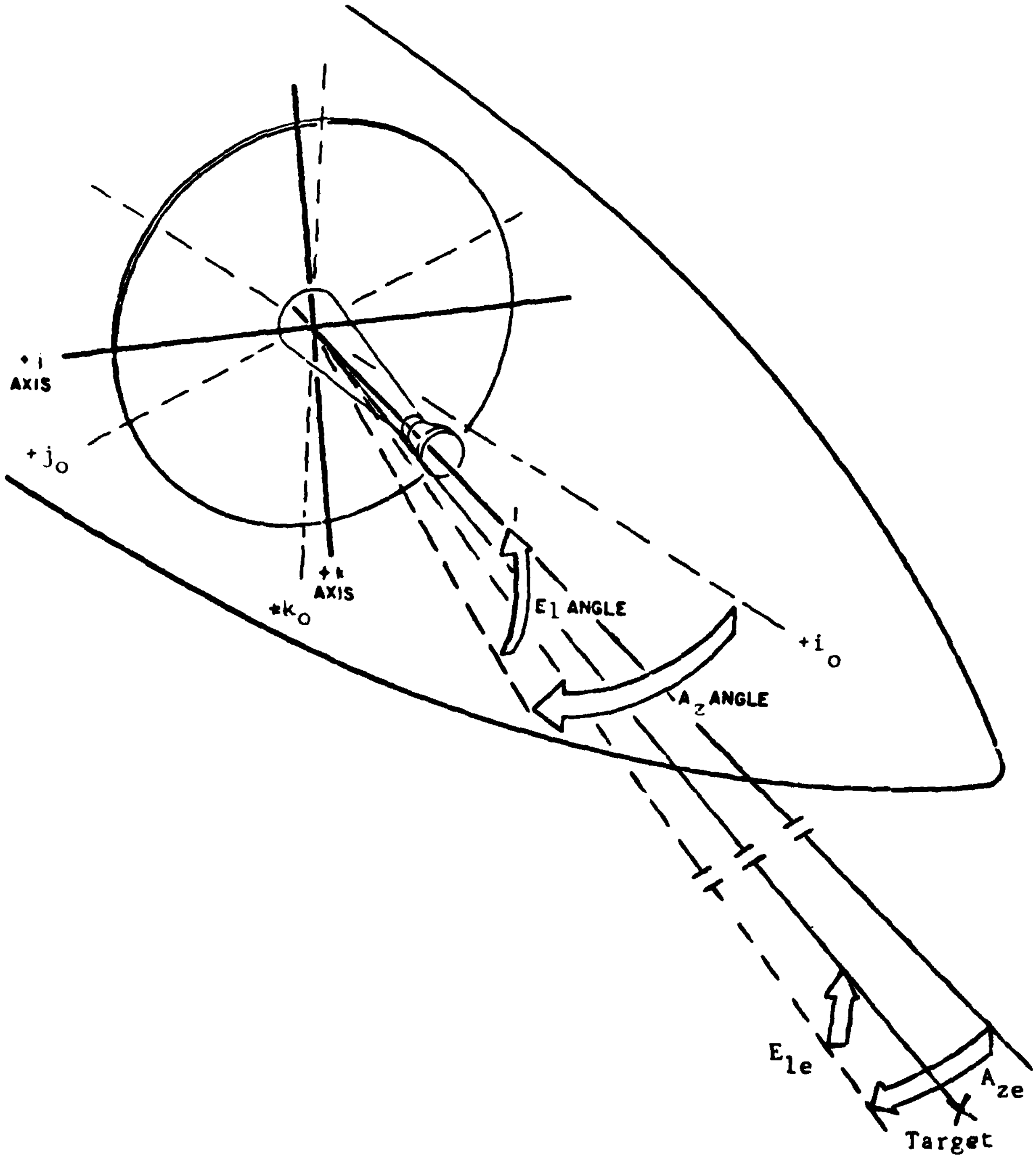


Figure 2-5 Antenna Tracker Reference, Coordinates, and Error Angles (A_{ZE} and E_{LE})(Adapted from T.O. 12P2-2APQ120-2-3-7)

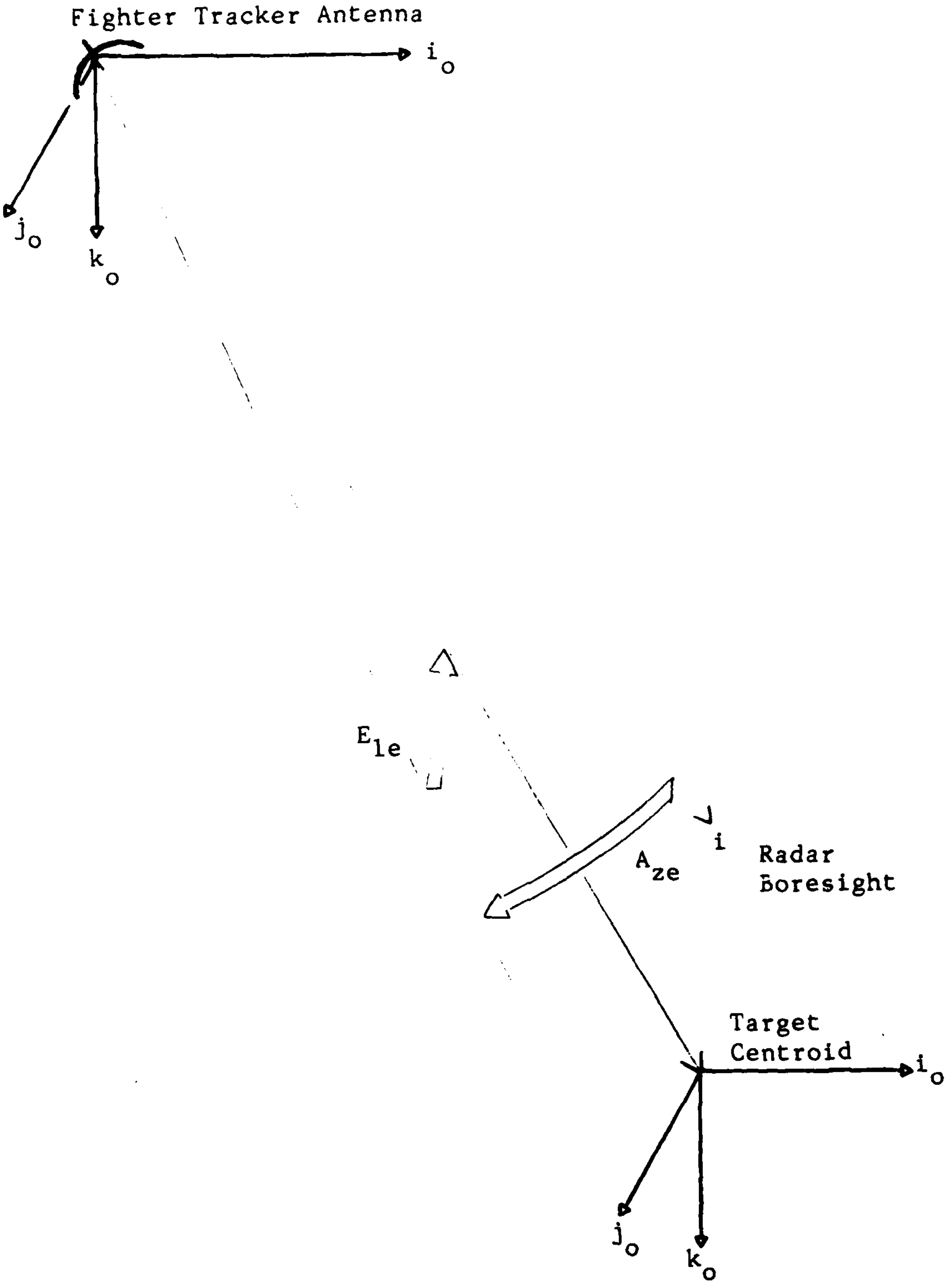


Figure 2-6 Filter Geometry

Next, Figure 2-5 illustrates Euler rotations of azimuth (A_z) and elevation (E_1) from the radar reference frame (i_o,j_o,k_o) to the radar tracker frame (i,j,k). Additionally, Euler rotations of azimuth error (A_{ze}) and elevation error (E_{1e}) around the radar tracker frame are illustrated to obtain the line-of sight (i_1,j_1,k_1) frame. Azimuth and elevation errors are included because of the radar measurement corruption noise discussed in Section 2.3.3. Figure 2-6 also illustrates these error angles.

Given the geometry, coordinate transformations are now defined. Starting with the geographic frame and multiplying by the rotation matrices in the order shown in Equation (2-4), the transformation from the geographic frame to the line-of-sight (1) frame is determined. Also note that the frames as well as the notation symbols, i.e., l,m,n for the aircraft frame, are included in Equation (2-4).

$$\begin{bmatrix} i\\ j\\ k\end{bmatrix}_1 = \underset{\substack{1\\ i_1,j_1,k_1}}{[E_{LE}]}\;[A_{ZE}]\;\underset{\substack{\text{TRACKER}\\ i,j,k}}{[E_L]}\;[A_Z]\;\underset{\substack{\text{RADAR}\\ i_o,j_o,k_o}}{[-2^\circ]}\;\underset{\substack{\text{AIRCRAFT}\\ l,m,n}}{[\phi]}\;[\theta]\;\underset{\substack{\text{SPACE}\\ x,y,z}}{[\Psi]}\;\underset{\substack{\text{GEOGRAPHIC}\\ N,E,D}}{\begin{bmatrix}N\\ E\\ D\end{bmatrix}} \qquad (2\text{-}4)$$

where,

$$[\Psi] = \begin{bmatrix} c\Psi & s\Psi & 0\\ -s\Psi & c\Psi & 0\\ 0 & 0 & 1\end{bmatrix},$$

$$[\theta] = \begin{bmatrix} c\theta & 0 & -s\theta \\ 0 & 1 & 0 \\ s\theta & 0 & c\theta \end{bmatrix},$$

$$[\phi] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & s\phi \\ 0 & -s\phi & c\phi \end{bmatrix},$$

$$[-2^\circ] = \begin{bmatrix} c(-2^\circ) & 0 & -s(-2^\circ) \\ 0 & 1 & 0 \\ s(-2^\circ) & 0 & c(-2^\circ) \end{bmatrix} = \begin{bmatrix} c2^\circ & 0 & s2^\circ \\ 0 & 1 & 0 \\ -s2^\circ & 0 & c2^\circ \end{bmatrix},$$

$$[A_Z] = \begin{bmatrix} cA_Z & sA_Z & 0 \\ -sA_Z & cA_Z & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$[E_L] = \begin{bmatrix} cE_L & 0 & -sE_L \\ 0 & 1 & 0 \\ sE_L & 0 & cE_L \end{bmatrix},$$

$$[A_{ZE}] = \begin{bmatrix} cA_{ZE} & sA_{ZE} & 0 \\ -sA_{ZE} & cA_{ZE} & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$[E_{LE}] = \begin{bmatrix} cE_{LE} & 0 & -sE_{LE} \\ 0 & 1 & 0 \\ sE_{LE} & 0 & cE_{LE} \end{bmatrix},$$

Ψ= Aircraft heading angle,

θ= Aircraft pitch angle,

ϕ= Aircraft roll angle,

-2°= Radar offset from aircraft reference (i versus i_o),

A_Z= Radar azimuth angle,

E_L= Radar elevation angle,

A_{ZE}= Azimuth error angle between radar and 1 frames,

E_{LE}= Elevation error angle between radar anu 1 frames,

N= North,

E= East,

D= Down,

c= cosine, and

s= sine

To reverse this process, i.e., to obtain N,E, and D from the 1 frame or i_1, j_1, and k_1, Equation (2-5) is employed.

$$\begin{bmatrix} N \\ E \\ D \end{bmatrix} = [\Psi]^T[\Theta]^T[\phi]^T[-2^\circ]^T[A_Z]^T[E_L]^T[A_{ZE}]^T[E_{LE}]^T\begin{bmatrix} i_1 \\ j_1 \\ k_1 \end{bmatrix} \quad (2\text{-}5)$$

where $[\cdot]^T$ is the transpose of $[\cdot]$. (Note that $[\cdot]^T$ can be used instead of $[\cdot]^{-1}$ (inverse) because the transformation matrices are orthogonal and the transpose equals the inverse.) Since the tracker is space stabilized, it is possible to determine the tracker attitude with respect to an non-rotating reference frame provided by the tracker's rate gyros and resolvers. The tracker frame is a fixed distance, d, from the fighter center of gravity (a Cartesian reference frame) and rotates about the center of gravity. Figures 2-1 and 2-4 illustrate the geometry of the tracker. The tracker frame orientation is obtained from Euler angle rotations Θ and ϕ, the two degree rotation factor, A_Z, and E_L, respectively, if the xyz is considered the reference frame. Denoting xyz frame as the reference frame I, and the tracking frame as T, the coordinate transformation matrix $C^{T/I}$ is defined as the transformation matrix from I coordinates into T coordinates

$$C^{T/I} = [E_L]\ [A_Z]\ [-2°]\ [\phi]\ [\theta] \qquad (2\text{-}6)$$

Further, denoting the line of sight frame as 1, the transformation from the tracker frame to the 1 frame is:

$$C^{1/T} = [E_{LE}]\ [A_{ZE}] \qquad (2\text{-}7)$$

Finally, the transformation from the reference frame to the 1 frame is:

$$C^{1/I} = C^{1/T}\ C^{T/I} \qquad (2\text{-}8)$$

2.4.3 State Equations and State Transition Matrix

Cartesian components of relative position, total velocity, and total target acceleration are used in describing the tracking state equations. The states are as follows (see Figure 2-7):

x_1 = relative x distance between the tracker reference frame and the target c.g.

x_2 = total target x velocity coordinatized in the tracker reference frame.

x_3 = total target x acceleration coordinatized in the tracker reference frame.

x_4 = relative y distance between the tracker reference frame and the target c.g.

x_5 = total target y velocity coordinatized in the tracker reference frame.

x_6 = total target y acceleration coordinatized in the tracker reference frame.

x_7 = relative z distance between the tracker reference frame and the target c.g.

x_8 = total target z velocity coordinatized in the tracker reference frame.

x_9 = total target z acceleration coordinatized in the tracker reference frame.

The above description agrees with the geometry established in Sections 2.2 and 2.4.2. Also note that the realtive position states forces the magnitudes of the position state estimates to a minimum when compared to total target position states. This limits a scaling problem between state estimates that can develop when total target position is used in the formation of the state vector.

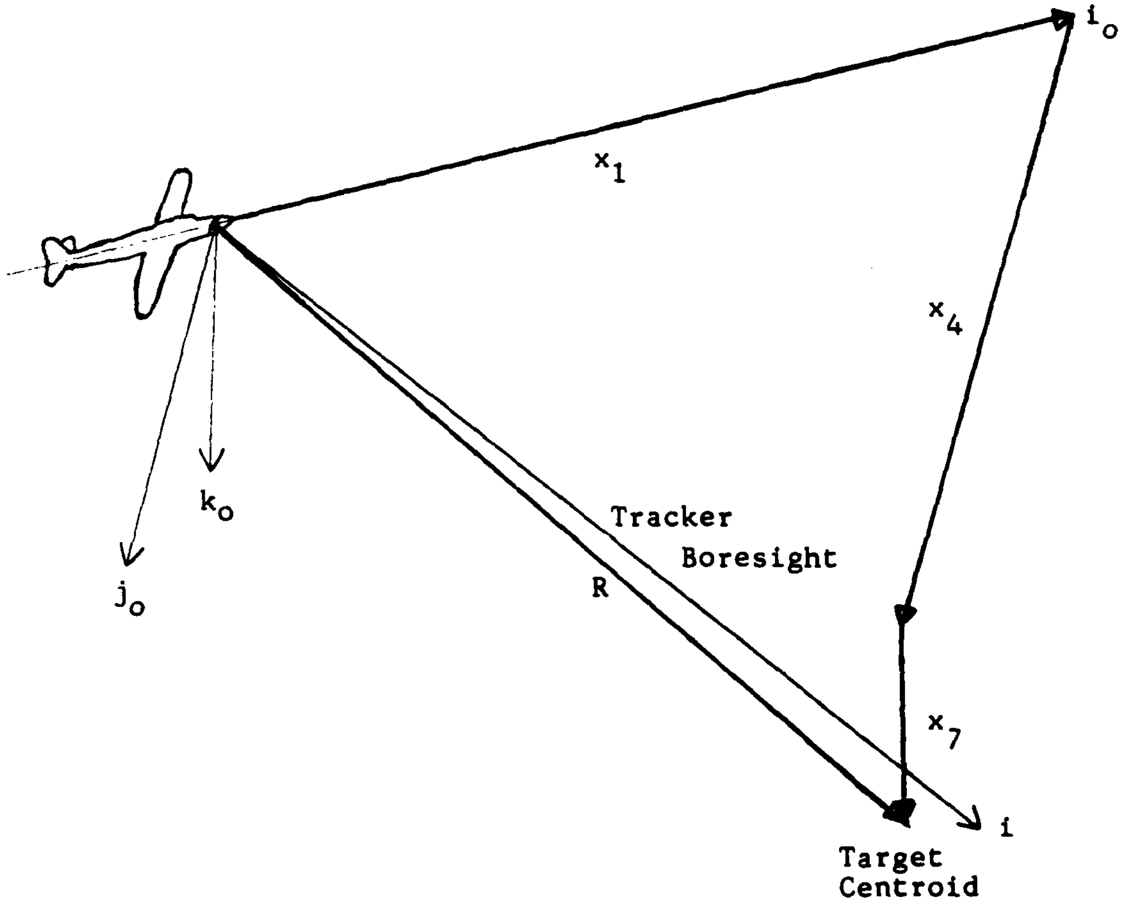


Figure 2-7 Fighter to Target Position States

As discussed in Section 2.2, a linear system dynamics model is desired. A first order Gauss-Markov acceleration model is selected because it allows the system model to remain linear. Higher order models such as a constant turn rate acceleration model (see (13) and (14) for further explanation) result in a nonlinear system dynamics model. Thus, the first order Gauss-Markov acceleration model yields the state space model:

$$\begin{aligned}
\dot{x}_1(t) &= x_2(t) - v_{fx}(t) \\
\dot{x}_2(t) &= x_3(t) \\
\dot{x}_3(t) &= (-1/\tau_x)x_3(t) + w_x(t) \\
\dot{x}_4(t) &= x_5(t) - v_{fy}(t) \\
\dot{x}_5(t) &= x_6(t) \\
\dot{x}_6(t) &= (-1/\tau_y)x_6(t) + w_y(t) \\
\dot{x}_7(t) &= x_8(t) - v_{fz}(t) \\
\dot{x}_8(t) &= x_9(t) \\
\dot{x}_9(t) &= (-1/\tau_z)x_9(t) + w_z(t)
\end{aligned} \qquad (2\text{-}9)$$

where,

$v_{fx,y,z}$= velocity of fighter in x,y, and z.

$\tau_{x,y,z}$ = time constants, tau, for acceleration models.

$w_{x,y,z}$ = dynamic noise driving the acceleration random process for x_3, x_6, and x_9, respectively.

In matrix form, after dropping the time argument, the above equations are represented as:

$$\dot{\underline{x}} = \left[\begin{array}{ccc|ccc|ccc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1/\tau_x & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1/\tau_y & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1/\tau_z \end{array}\right] \underline{x} + \left[\begin{array}{ccc} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right] \begin{bmatrix} v_{fx} \\ v_{fy} \\ v_{fz} \end{bmatrix} + \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{array}\right] \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} \tag{2-10}$$

which is of the form

$$\dot{\underline{x}} = \underline{F}\underline{x} + \underline{B}\underline{u} + \underline{G}\underline{w} \qquad (2\text{-}11)$$

The state transition matrix, $\underline{\Phi}(t,t_o)$, for this time-invariant system model, is defined as

$$\underline{\Phi}(t,t_o) = \pounds^{-1}[s\underline{I} - \underline{F}]^{-1} \qquad (2\text{-}12)$$

where,

$\pounds^{-1}$ = inverse Laplace transform

s = Laplace operator

$\underline{I}$ = identity matrix

$\underline{F}$ = as defined in Equations (2-10) and (2-11)

Performing this operation results in a block diagonal matrix of the form

$$\underline{\Phi}(t_{i+1},t_i) = \underline{\Phi}(t) = \left[\begin{array}{c|c|c} \underline{\Phi}_x & 0 & 0 \\ \hline 0 & \underline{\Phi}_y & 0 \\ \hline 0 & 0 & \underline{\Phi}_z \end{array}\right] \qquad (2\text{-}13)$$

where, each element, $\underline{\Phi}_i$, $i=x,y,z$, is a 3x3 matrix with

$$\underline{\Phi}_i = \begin{bmatrix} 1 & \Delta t & \tau_i^2[(1/\tau_i)\Delta t - 1 + e^{-(1/\tau_i)\Delta t}] \\ 0 & 1 & \tau_i[1 - e^{-(1/\tau_i)\Delta t}] \\ 0 & 0 & e^{-(1/\tau_i)\Delta t} \end{bmatrix}_{i=x,y,z} \qquad (2\text{-}14)$$

Δt = the time interval (sample period) from one update until the next, and

τ_i = acceleration time constant, i=x,y,z.

2.4.4 State Equations Translated to the C.G

As discussed in Section 2.2, the antenna tracker reference frame or radar reference frame is the primary reference frame used in this thesis. The radar reference frame is offset a distance, d, from and rotates about the c.g. reference frame. As such, it is sometimes desirable to perform calculations in the c.g. Cartesian frame since Coriolis and centripetal acceleration terms equal zero. Although not used explicitly in this thesis, this section is included for completeness.

Defining $\underline{d}$ as (body or aircraft frame)

$$\underline{d} = \begin{bmatrix} d \\ 0 \\ 0 \end{bmatrix}_{l,m,n} \tag{2-15}$$

and describing $\underline{d}$ in x,y,z (c.g. Cartesian) coordinates gives

$$\underline{d} = \begin{bmatrix} d(c\theta) \\ 0 \\ -d(s\theta) \end{bmatrix}_{x,y,z} = [\theta]^T[\phi]^T \begin{bmatrix} d \\ 0 \\ 0 \end{bmatrix}_{lmn} \tag{2-16}$$

Then the position states at the c.g. are:

$$\begin{aligned} X_{1cg} &= X_1 + d(c\theta) \\ X_{4cg} &= X_4 \\ X_{5cg} &= X_7 - d(s\theta) \end{aligned} \tag{2-17}$$

Next, $\underline{w}$ is defined in the c.g. Cartesian frame as:

$$\underline{w} = \begin{bmatrix} \dot{\phi} \\ \dot{\Theta} \\ \dot{\Psi} \end{bmatrix}_{x,y,z} \qquad (2\text{-}18)$$

The velocity states in the c.g. frame are found by forming the cross product $\underline{w}x\underline{d}$ and adding it to the velocity in the tracker state (by the Corolis theorem)

$$(\underline{w}_{x,y,z}) \times (\underline{d}_{x,y,z}) = \begin{bmatrix} -\dot{\Theta}d(s\Theta) \\ \dot{\Psi}d(c\Theta) + \dot{\phi}d(s\Theta) \\ -\dot{\Psi}d(c\Theta) \end{bmatrix} \qquad (2\text{-}19)$$

The velocity states at the c.g. are:

$$\begin{aligned} X_{2cg} &= X_2 - \dot{\Theta}d(s\Theta) \\ X_{5cg} &= X_5 + \dot{\Psi}d(c\Theta) + \dot{\phi}d(s\Theta) \\ X_{8cg} &= X_8 - \dot{\Psi}d(c\Theta) \end{aligned} \qquad (2\text{-}20)$$

Finally, the ownship acceleration vector $(a_{fx}, a_{fy}, a_{fz})^T$ is determined from ownship accelerometers which are located near the aircraft c.g. The total target acceleration (radar frame) is described by states X_3, X_6, and X_9. The relative acceleration can be found by simply subtracting total inertial target acceleration from ownship acceleration.

Thus, the acceleration terms at the radar tracker must be transformed to the inertial frame. This is accomplished by the following formula (by the Corolis theorem):

$$\underline{a}_{x,y,z} = \underline{a}_{tracker} + \dot{\underline{w}}\text{x}\underline{d} + \underline{w}\text{x}(\underline{w}\text{x}\underline{d}) \qquad (2\text{-}21)$$

Expressing the last two terms of this equation in inertial terms and evaluating results in:

$$\underline{a}_{c.g.} = \begin{bmatrix} X_3 - \ddot{\theta}d(s\theta) - \dot{\theta}\dot{\psi}(dc\theta) - \dot{\psi}(\dot{\phi}(ds\theta) + \dot{\psi}d(c\theta)) \\ X_6 + \ddot{\psi}d(c\theta) + \ddot{\phi}(ds\theta) - \dot{\psi}(\dot{\theta}(ds\theta) - \dot{\phi}(dc\theta)) \\ X_9 - \ddot{\psi}(dc\theta) + \dot{\phi}(\dot{\phi}(ds\theta) + \dot{\psi}(dc\theta)) + \dot{\theta}^2(ds\theta) \end{bmatrix} \qquad (2\text{-}22)$$

2.5 Measurement Models

2.5.1 Introduction

As discussed in Section 2.2 and the last section, the F-4E/G radar is located in the nose of the aircraft, which rotates about the aircraft c.g. during a maneuver. Since the distance, d, between the c.g. and the radar is fixed, it is possible to account for both the translation effects from the c.g. to the radar unit and rotational effects about the c.g. The effects do not impact on the measurement model directly. The measurement model is referenced to the tracker frame in order to compare actual measurements directly to "measurements" from the measurement model, i.e., to allow simple residual generation in the eventual filter.

2.5.2 Measurement Model (Radar Reference Frame)

Actual noise-corrupted measurement realizations are provided by the radar as range (R), range rate (R_{DOT}), azimuth angle (A_Z), elevation angle (E_L), azimuth rate (w_k or A_{ZDOT}), and elevation rate (w_j or E_{LDOT}). The Kalman filter requires discrete-time measurements modeled in terms of the states in order to form a residual (see Section 2.6). As discussed in Section 2.2, the measurement equations are nonlinear when modeled in a Cartesian frame. Figures 2-7, 2-8, and 2-9 illustrate the geometry.

The following are measurement equations based on the state space and the geometry defined.

For range, R (see Figure 2-7):

$$Z_1 = R = (X_1{}^2 + X_4{}^2 + X_7{}^2)^{1/2} + v_1 \qquad (2\text{-}23)$$

where v_1 is described statistically in Table II.4.

For range rate, R_{DOT}:

$$\begin{aligned} Z_2 &= R_{DOT} = \dot{R} \\ &= \frac{X_1(X_2 - v_{fx}) + X_4(X_5 - v_{fy}) + X_7(X_8 - v_{fz})}{(X_1{}^2 + X_4{}^2 + X_7{}^2)^{1/2}} + v_2 \qquad (2\text{-}24) \end{aligned}$$

where v_2 is described in Table II-4.

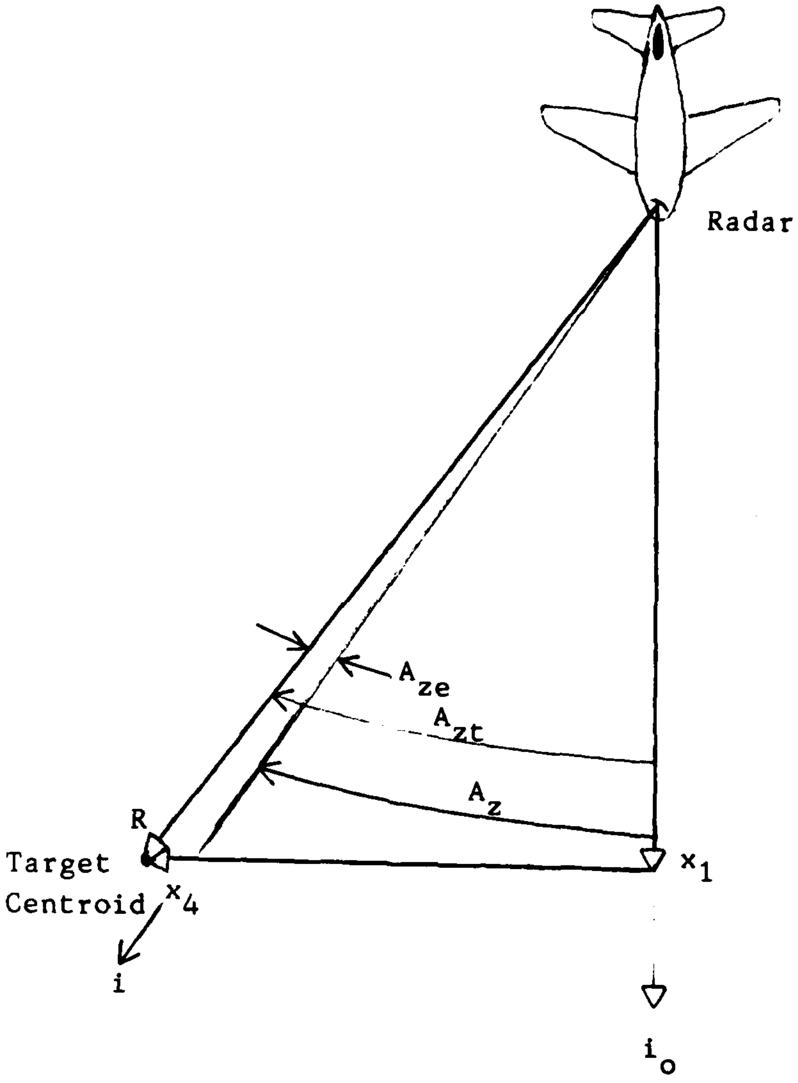


Figure 2-8 Azimuth (top view)

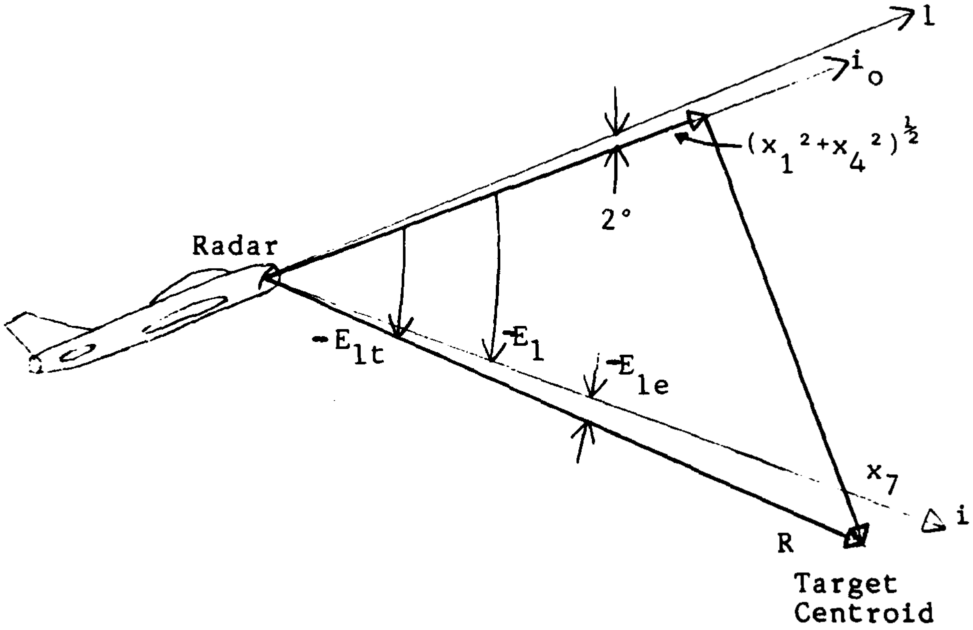


Figure 2-9 Elevation after Azimuth Rotation (side view)

For azimuth angle, A_Z, the total azimuth angle from the geometry (see Figure 2-8) is

$$A_{ZT} = A_Z + A_{ZE} \tag{2-25}$$

and,

$$A_{ZT} = \tan^{-1}(X_4/X_1) \tag{2-26}$$

where A_{ZE} is the error angle between the radar boresight and the line-of-sight (l) vector projection onto the azimuth plane. This angle is typically small (8:3-361) and is modeled as zero mean white noise. Thus,

$$Z_3 = A_Z = \tan^{-1}(X_4/X_1) + v_3 \tag{2-27}$$

where v_3 includes noise from Table II-4 plus the azimuth error.

For elevation angle, E_L, the total elevation angle from the geometry (see Figure 2-9) is

$$E_{LT} = E_L + E_{LE} \tag{2-28}$$

and,

$$E_{LT} = -\tan^{-1} \frac{X_7}{(X_1{}^2 + X_4{}^2)^{1/2}} \tag{2-29}$$

where E_{LE} is the error angle between the radar boresight and the l vector projection onto the elevation plane. Again, this angle is typically small and is modeled as zero mean white noise. Thus,

$$Z_4 = E_L = -\tan^{-1} \frac{X_7}{(X_1{}^2 + X_4{}^2)^{1/2}} + v_4 \qquad (2\text{-}30)$$

where v_4 includes noises from Table II-4 plus the elevation error term.

For azimuth rate, w_k or A_{ZDOT}:

$$\begin{aligned} Z_5 &= w_k = A_{ZDOT} = \dot{A}_Z \\ &= \frac{X_1(X_5-v_{fy}) - (X_2-v_{fx})X_4}{X_1{}^2 + X_4{}^2} + v_5 \qquad (2\text{-}31) \end{aligned}$$

where v_5 is described in Table II-4.

Finally, for elevation rate, w_j or ELDOT:

$$\begin{aligned} Z_6 &= w_j = E_{LDOT} = \dot{E}_L \\ &= -\frac{(X_8-v_{fz})(X_1{}^2 + X_4{}^2) - X_7[X_1(X_2-v_{fx}) + X_4(X_5-v_{fy})]}{(X_1{}^2 + X_4{}^2 + X_7{}^2)(X_1{}^2 + X_4{}^2)^{1/2}} + v_6 \end{aligned}$$

$$(2\text{-}32)$$

where v_6 is described in Table II-4.

For Z_3 and Z_4, the "azimuth and elevation angle errors can be as large as 0.5 degree" (8:3-361). However, the truth model (see Section 4-2) outputs significantly larger angle errors, as large as 2.35 degrees when the fighter is maneuvering. Concurrent with the angle errors, azimuth and elevation angle rate errors as large as 7.2 degrees per second are observed. OO-ALC/MMECB indicates that the observed truth model errors are realistic for the F-4E/G aircraft (15). The large angle errors and angle rate errors are attributed to the radar dynamics. Treating 2.35 degrees as a three-sigma value and modeling the error as zero-mean white Gaussian noise results in a one-sigma value of 0.783 degrees. Adding the nominal one-sigma noise value from Table II-4 results in an overall one-sigma value of approximately one degree. Similarily, the angle rate error terms observed plus the nominal value results in a one-sigma value of approximately 3.1 degrees per second. From simulation runs conducted in Chapter V, a one-sigma of 3.1 is determined to be too large (by studying measurement residual plots) and consequently reduced to 1.26 degrees per second. Additionally, the range and range rate terms of $\underline{R}_{nom}$ are considered too small and are increased as shown in Equation (2-33).

$$\underline{R} = \begin{bmatrix} 2500 & 0 & 0 & 0 & 0 & 0 \\ 0 & 625 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3.0\text{x}10^{-4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 3.0\text{x}10^{-4} & 0 & 0 \\ 0 & 0 & 0 & 0 & 4.9\text{x}10^{-4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 4.9\text{x}10^{-4} \end{bmatrix} \quad (2\text{-}33)$$

Summarizing the measurement equations yields

$$\underline{z}(t_i) = \underline{h}[\underline{x}(t_i)] + \underline{v}$$

$$= \begin{bmatrix} (X_1{}^2 + X_4{}^2 + X_7{}^2)^{1/2} \\[2ex] \dfrac{X_1(X_2-v_{fx}) + X_4(X_5-v_{fy}) + X_7(X_8-v_{fz})}{(X_1{}^2 + X_4{}^2 + X_7{}^2)^{1/2}} \\[2ex] \tan^{-1}(X_4/X_1) \\[2ex] -\tan^{-1} \dfrac{X_7}{(X_1{}^2 + X_4{}^2)^{1/2}} \\[2ex] \dfrac{X_1(X_5-v_{fy}) - (X_2-v_{fx})X_4}{X_1{}^2 + X_4{}^2} \\[2ex] -\dfrac{(X_8-v_{fz})(X_1{}^2 + X_4{}^2) - X_7[X_1(X_2-v_{fx}) + X_4(X_5-v_{fy})]}{(X_1{}^2 + X_4{}^2 + X_7{}^2)(X_1{}^2 + X_4{}^2)^{1/2}} \end{bmatrix} + \begin{bmatrix} v_1 \\[2ex] v_2 \\[2ex] v_3 \\[2ex] v_4 \\[2ex] v_5 \\[2ex] v_6 \end{bmatrix} \tag{2-34}$$

2.6 Truth Model

The truth model is comprised of the radar servo model and associated true target and filter states derived from a trajectory generation program (see Chapter IV). For the purpose of evaluating the filter performance, it is only necessary to generate data points of relative positions, total target velocities, and total target accelerations at discrete times, t_i. For the purpose of computer simulation, it is not necessary for the fighter to follow the target, only to track it. Thus the information from the Kalman filter is not fed back to the simulated aircraft (as it will be to an actual aircraft to determine control inputs), simulated filter outputs are only compared to the true states to generate a statistical analysis. This is presented pictorially in Figure 2-10.

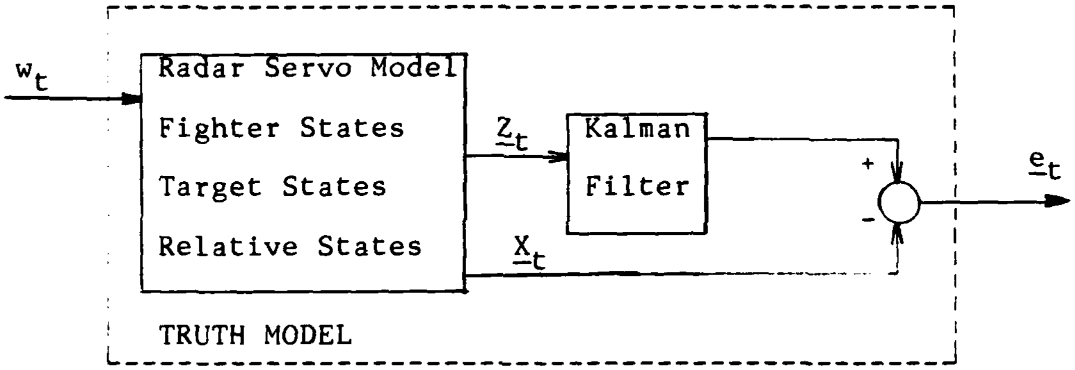


Figure 2-10 Performance Evaluation of the Kalman Filter

III. EXTENDED KALMAN FILTER DESIGN

3.1 Introduction

The purpose of this chapter is to derive appropriate models for various Kalman filters for theoretical study and possible implementation on the F-4E/G. It is shown in Chapter II that the selection of a Cartesian coordinate frame resulted in nonlinear measurement equations. Thus a linear Kalman filter implementation can not be employed. A linearized, or perturbation Kalman filter can be used but the nominal trajectory for an air-to-air tracking scenario is not known a priori. Choosing an arbitrary nominal path results in large perturbations and can cause filter divergence. Moreover, an extended Kalman filter (EKF) has the same basic form as the linear and linearized Kalman filter except that $\underline{h}[\hat{\underline{x}}(t_i^-),t_i]$ is used to form the residual, rather than $(\underline{h}[x_{nom},t_i] + \underline{H}\ \underline{x})$. An EKF uses the state estimates to relinearize the filter about a new reference state trajectory each time a new state estimate vector is calculated. The linearization is a first order approximation of a Taylor series expansion about the estimate of the state vector. Higher order filters such as modified truncated second order filters and modified Gaussian second order filters provide performance enhancement over the EKF by reducing estimate bias but with added computational complexity (16:221-223). Because of the limited memory of the F-4E/G fire control system, the filter employed in this research is limited to a first order filter without bias correction. As it is shown in Chapter V, bias correction is not necessary because the system displays essentially zero-mean performance when the fighter is not maneuvering.

As explained in Section 1.2, the present F-4E/G target estimation filter does not accurately estimate target parameters during an F-4 maneuver. It may be the current Wiener-Hopf filter is improperly tuned. An EKF that is properly tuned may in itself provide adequate target estimates. Alternately, an off-line adaptive EKF is studied for its effect on response. An adaptive filter varies filter parameters (as gain and/or $\underline{Q}$ and $\underline{R}$) in response to a given decision used to change the weighting of measurement for incorrect system modeling. "Off-line" adaptive estimation is a process of varying the filter tuning parameters, and thus gain, to alter filter performance based on a priori information. Off-line adaptive estimation provides a baseline of performance that an on-line adaptive estimator can theoretically approach. Since the acceleration of the target is random with respect to the fighter, it may be necessary to make adaptive adjustments during periods of detected acceleration to correct for changed target behavior. Furthermore, as it is shown later, filter performance is the worst when the fighter maneuvers, and therefore, it probably is necessary to make adaptive changes during a fighter maneuver.

There has been considerable research done on using a variation of a Kalman filter for solving precision pointing and tracking problem. Appendix A contains abstracts from selected theses and published articles which studied Kalman filters. Thus the problem has been studied before and the application of the Kalman filter is well established.

3.2 EKF Design

In this section, EKF equations are presented without derivation. Then the required $\underline{f}$ and $\underline{h}$ vectors and corresponding $\underline{F}$ and $\underline{H}$ partial derivative matrices for use in the EKF design are derived. Next, $\underline{Q}$ is calculated based on a $\mathcal{T}_{x,y,z}$ for a fighter type aircraft. This is followed by evaluation of initial conditions, $\hat{\underline{x}}_o$ and $\underline{P}_o$. These steps constitute the design and allow for a follow-on system evaluation.

3.2.1 EKF Equations

Using page 44 of Reference 16 as a guide, the EKF equations are presented without proof. From Equations (2-10) and (2-11) the dynamics model of interest is

$$\dot{\underline{x}}(t) = \underline{f}[\underline{x}(t),\underline{u}(t),t] + \underline{G}(t)\underline{w}(t) \qquad (3\text{-}1)$$

where:

$\underline{x}(t)$ is the n-state filter vector [$\underline{x}(t_o)$ is modeled as a Gaussian random vector with mean $\hat{\underline{x}}_o$ and covariance $\underline{P}_o$]

$\underline{f}[\underline{x}(t),\underline{u}(t),t]$ is the filter dynamics vector

$\underline{u}(t)$ is a r-vector of known input functions

$\underline{G}(t)$ is a n-by-s noise input matrix, and

$\underline{w}(t)$ is a zero-mean white Gaussian s-vector process with strength $\underline{Q}(t)$,

$$E[\underline{w}(t)\underline{w}^T(t+\tau)] = \underline{Q}(t)\delta(\tau) \tag{3-2}$$

and independent of $\underline{x}(t_o)$.

From Equation (2-34), the available discrete-time measurements are modeled as the m-vector process $\underline{z}(t_i)$

$$\underline{z}(t_i) = \underline{h}[\underline{x}(t_i),t_i] + \underline{v}(t_i) \tag{3-3}$$

where,

$\underline{v}(t_i)$ is a zero-mean white Gaussian m-vector process with covariance $\underline{R}(t_i)$, independent of $\underline{x}(t_o)$ and $\underline{w}(t)$,

$\underline{h}[\underline{x}(t_i),t_i]$ is the measurement model vector.

The measurement update incorporates measurements $\underline{z}(t_i,w_j) = \underline{z}_i$ by

$$\underline{K}(t_i)=[\underline{P}(t_i^-)\underline{H}^T[t_i;\hat{\underline{x}}(t_i^-)]] \\ [\underline{H}[t_i;\hat{\underline{x}}(t_i^-)]\underline{P}(t_i^-)\underline{H}^T[t_i;\hat{\underline{x}}(t_i^-)] + \underline{R}(t_i)]^{-1} \tag{3-4}$$

$$\hat{\underline{x}}(t_i^+) = \hat{\underline{x}}(t_i^-) + \underline{K}(t_i)[\underline{z}_i - \underline{h}[\hat{\underline{x}}(t_i^-),t_i]] \tag{3-5}$$

$$\underline{P}(t_i^+) = \underline{P}(t_i^-) - \underline{K}(t_i)\underline{H}[t_i;\hat{\underline{x}}(t_i^-)]\underline{P}(t_i^-) \tag{3-6}$$

where,

$\underline{K}(t_i)$ is the Kalman filter gain,

$\underline{H}[t_i;\hat{\underline{x}}(t_i^-)]$ is the m-by-n partial derivative matrix

$$\underline{H}[t_i;\hat{\underline{x}}(t_i^-)] \triangleq \left.\frac{\partial \underline{h}[\underline{x},t_i]}{\partial \underline{x}}\right|_{\underline{x}=\hat{\underline{x}}(t_i^-)} \quad \text{,and} \qquad (3\text{-}7)$$

$\underline{P}(t_i)$ is the filter-computed conditional covariance matrix; "+" specifies covariance is referenced to time just after update time t_i, "-" specifies just before time t_i.

The estimate is propagated forward to the next sample time t_{i+1} by integrating

$$\dot{\hat{\underline{x}}}(t/t_i) = \underline{f}[\hat{\underline{x}}(t/t_i),\underline{u}(t),t] \qquad (3\text{-}8)$$

$$\dot{\underline{P}}(t/t_i) = \underline{F}[t;\hat{\underline{x}}(t/t_i)]\underline{P}(t/t_i) + \underline{P}(t/t_i)\underline{F}^T[t;\hat{\underline{x}}(t/t_i)] + G(t)Q(t)G^T(t) \qquad (3\text{-}9)$$

from time t_i to t_{i+1}, where (t/t_i) represents time t for t as an element of $[t_i,t_{i+1})$ (given measurements through time t_i). Additionally, the initial conditions are provided in Equations (3-5) and (3-6) as

$$\hat{\underline{x}}(t_i/t_i) = \hat{\underline{x}}(t_i^+) \qquad (3\text{-}10)$$

$$\underline{P}(t_i/t_i) = \underline{P}(t_i^+) \qquad (3\text{-}11)$$

For the first interval, time t_o to t_1, the initial conditions are $\hat{\underline{x}}_o$ and $\underline{P}_o$. In Equation (3-9), $\underline{F}[t;\hat{\underline{x}}(t/t_i)]$ is the n-by-n partial derivative matrix:

$$\underline{F}[t;\underline{x}(t/t_i)] \triangleq \left.\frac{\partial \underline{f}[\underline{x},\underline{u}(t),t]}{\partial \underline{x}}\right|_{\underline{x}=\hat{\underline{x}}(t/t_i)} \quad (3\text{-}12)$$

for all t in the interval $[t_i,t_{i+1})$. As discussed in Section 2.2, using a Cartesian coordinate frame for the dynamics models of Chapter 2 results in $\underline{f}(=\underline{Fx})$ being linear and time-invariant.

Finally, by integrating Equations (3-8) and (3-9) to the next sample time, $\hat{\underline{x}}(t_{i+1}^-)$ and $\underline{P}(t_{i+1}^-)$ are defined as

$$\hat{\underline{x}}(t_{i+1}^-) = \hat{\underline{x}}(t_{i+1}/t_i) \quad (3\text{-}13)$$

$$\underline{P}(t_{i+1}^-) = \underline{P}(t_{i+1}/t_i) \quad (3\text{-}14)$$

for use in the next measurement update.

The theory presented in this section is applied in a computer-aided design package called SOFE (see Section 4-3). However, as explained in Section 2.4.3, a state transition matrix approach can be used to simplify the above propagation (see Section 4.4).

3.2.2 Evaluation of $\underline{H}[t_i;\hat{\underline{x}}(t_i^-)]$ and $\underline{F}[t;\hat{\underline{x}}(t/t_i)]$

For the dynamics measurement model in Chapter II, Equation (2-34), $\underline{H}[t_i;\hat{\underline{x}}(t_i^-)]$ is derived. Next, for the state vector differential equation, Equation (2-10), $\underline{F}[t;\hat{\underline{x}}(t/t_i)]$ is calculated. The results are used in the EKF design.

For $\underline{H}[t_i;\hat{\underline{x}}(t_i^-)]$,

$$\underline{H}[t_i;\hat{\underline{x}}(t_i^-)] = \begin{bmatrix} H_{11} & 0 & 0 & H_{14} & 0 & 0 & H_{17} & 0 & 0 \\ H_{21} & H_{22} & 0 & H_{24} & H_{25} & 0 & H_{27} & H_{28} & 0 \\ H_{31} & 0 & 0 & H_{34} & 0 & 0 & 0 & 0 & 0 \\ H_{41} & 0 & 0 & H_{44} & 0 & 0 & H_{47} & 0 & 0 \\ H_{51} & H_{52} & 0 & H_{54} & H_{55} & 0 & 0 & 0 & 0 \\ H_{61} & H_{62} & 0 & H_{64} & H_{65} & 0 & H_{67} & H_{68} & 0 \end{bmatrix} \qquad (3\text{-}15)$$

where,

$$H_{11}=\frac{x_1}{(x_1^2+x_4^2+x_7^2)^{1/2}} \qquad (3\text{-}16)$$

$$H_{14}=\frac{x_4}{(x_1^2+x_4^2+x_7^2)^{1/2}} \qquad (3\text{-}17)$$

$$H_{17}=\frac{x_7}{(x_1^2+x_4^2+x_7^2)^{1/2}} \qquad (3\text{-}18)$$

$$H_{21}=\frac{\begin{array}{l}(x_2-v_{fx})(x_1^2+x_4^2+x_7^2)- \\ x_1[x_1(x_2-v_{fx})+x_4(x_5-v_{fy})+x_7(x_8-v_{fz})]\end{array}}{(x_1^2+x_4^2+x_7^2)^{3/2}} \qquad (3\text{-}19)$$

$$H_{22}=\frac{x_1}{(x_1^2+x_4^2+x_7^2)^{1/2}} \qquad (3\text{-}20)$$

$$H_{24}=\frac{\begin{array}{l}(x_5-v_{fy})(x_1^2+x_4^2+x_7^2)- \\ x_4[x_1(x_2-v_{fx})+x_4(x_5-v_{fy})+x_7(x_8-v_{fz})]\end{array}}{(x_1^2+x_4^2+x_7^2)^{3/2}} \qquad (3\text{-}21)$$

$$H_{25}=\frac{x_4}{(x_1^2+x_4^2+x_7^2)^{1/2}} \qquad (3\text{-}22)$$

$$H_{27}=\frac{(x_8-v_{fz})(x_1{}^2+x_4{}^2+x_7{}^2)-x_7[x_1(x_2-v_{fx})+x_4(x_5-v_{fy})+x_7(x_8-v_{fz})]}{(x_1{}^2+x_4{}^2+x_7{}^2)^{3/2}} \tag{3-23}$$

$$H_{28}=\frac{x_7}{(x_1{}^2+x_4{}^2+x_7{}^2)^{1/2}} \tag{3-24}$$

$$H_{31}=\frac{-x_4}{x_1{}^2+x_4{}^2} \tag{3-25}$$

$$H_{34}=\frac{x_1}{x_1{}^2+x_4{}^2} \tag{3-26}$$

$$H_{41}=\frac{x_1x_7}{(x_1{}^2+x_4{}^2+x_7{}^2)(x_1{}^2+x_4{}^2)^{1/2}} \tag{3-27}$$

$$H_{44}=\frac{x_4x_7}{(x_1{}^2+x_4{}^2+x_7{}^2)(x_1{}^2+x_4{}^2)^{1/2}} \tag{3-28}$$

$$H_{47}=\frac{-(x_1{}^2+x_4{}^2)^{1/2}}{x_1{}^2+x_4{}^2+x_7{}^2} \tag{3-29}$$

$$H_{51}=\frac{2x_1(x_2-v_{fx})x_4+(x_5-v_{fy})(x_4{}^2-x_1{}^2)}{(x_1{}^2+x_4{}^2)^2} \tag{3-30}$$

$$H_{52}=\frac{-x_4}{x_1{}^2+x_4{}^2} \tag{3-31}$$

$$H_{54}=\frac{-2x_1x_4(x_5-v_{fy})+(x_2-v_{fx})(x_4{}^2-x_1{}^2)}{(x_1{}^2+x_4{}^2)^2} \tag{3-32}$$

$$H_{55}=\frac{x_1}{x_1^2+x_4^2} \tag{3-33}$$

$$H_{61}=\frac{-[(x_1^2+x_4^2+x_7^2)(x_1^2+x_4^2)(2x_1(x_8-v_{fz})-(x_2-v_{fx})x_7)-[(x_8-v_{fz})(x_1^2+x_4^2)-x_7(x_1(x_2-v_{fx})+x_4(x_5-v_{fy}))][x_1]\cdot[2(x_1^2+x_4^2)+(x_1^2+x_4^2+x_7^2)]]}{(x_1^2+x_4^2+x_7^2)^2(x_1^2+x_4^2)^{3/2}} \tag{3-34}$$

$$H_{62}=\frac{x_1x_7}{(x_1^2+x_4^2+x_7^2)(x_1^2+x_4^2)^{1/2}} \tag{3-35}$$

$$H_{64}=\frac{-[(x_1^2+x_4^2+x_7^2)(x_1^2+x_4^2)(2x_4(x_8-v_{fz})-(x_5-v_{fy})x_7)-[(x_8-v_{fz})(x_1^2+x_4^2)-x_7(x_1(x_2-v_{fx})+x_4(x_5-v_{fy}))][x_4]\cdot[2(x_1^2+x_4^2)+(x_1^2+x_4^2+x_7^2)]]}{(x_1^2+x_4^2+x_7^2)^2(x_1^2+x_4^2)^{3/2}} \tag{3-36}$$

$$H_{65}=\frac{x_4x_7}{(x_1^2+x_4^2+x_7^2)(x_1^2+x_4^2)^{1/2}} \tag{3-37}$$

$$H_{67}=\frac{(x_1^2+x_4^2+x_7^2)(x_1^2+x_4^2)(x_1(x_2-v_{fx})+x_4(x_5-v_{fy}))+[(x_8-v_{fz})(x_1^2+x_4^2)-x_7(x_1(x_2-v_{fx})+x_4(x_5-v_{fy}))]\cdot[2][x_7][(x_1^2+x_4^2)]}{(x_1^2+x_4^2+x_7^2)^2(x_1^2+x_4^2)^{3/2}} \tag{3-38}$$

$$H_{68}=\frac{-(x_1^2+x_4^2)}{(x_1^2+x_4^2+x_7^2)(x_1^2+x_4^2)^{1/2}} \tag{3-39}$$

$\underline{F}[t;\hat{\underline{x}}(t/t_i)]$ is identical to $\underline{F}$ derived in Equations (2-10) and (2-11).

3.2.3 EKF Noise Strengths

The measurement noises are derived in Chapter II and are summarized in Equation (2-34). The strengths of the dynamic driving noises w_x, w_y, and w_z (see Equations (2.9) and (2.10)) represent the uncertainties in the choices of τ_x, τ_y, and τ_z for the acceleration state equations. In matrix form, Equation (2-2) becomes

$$\underline{Q}(t) = \begin{bmatrix} Q_1 & 0 & 0 \\ 0 & Q_2 & 0 \\ 0 & 0 & Q_3 \end{bmatrix} \tag{3-40}$$

The initial choice of values for Q_1, Q_2, and Q_3 are not critical since $\underline{Q}$ values are changed during the filter tuning process to obtain the best filter performance (see Chapter V). On the other hand, since $\underline{R}$ is based on the available measurements, $\underline{R}$ is not normally varied in this study. Thus, the initial value of $\underline{Q}$ is selected as follows

$$Q_{1,2,3} = 2\sigma^2/\tau_{x,y,z} \tag{3-41}$$

Using a τ equal to 0.5 seconds, which is typical for a fighter type aircraft, a σ equal to 3.0 g's or 96.6 ft/sec^2 (17), and assuming symmetry in all axes (not unrealistic because target acceleration is random to the fighter and is as probable to occur in one axis as another) results in

$$\underline{Q}(t) = \begin{bmatrix} 37{,}325 & 0 & 0 \\ 0 & 37{,}325 & 0 \\ 0 & 0 & 37{,}325 \end{bmatrix} \qquad (3\text{-}42)$$

where the units of $\underline{Q}$ are feet^2/seconds^5.

3.2.4 Initial Conditions, $\hat{\underline{x}}_o$ and $\underline{P}_o$

As discussed in Section 3.2, $\underline{x}(t_o)$ is a Gaussian random variable with mean $\hat{\underline{x}}_o$ and covariance $\underline{P}_o$. Conceptially, $\hat{\underline{x}}_o$ can be obtained from the first measurement. (For simulation, these are the states at the start of the simulation; for actual implementation, the first measurement after radar lock-on.) The covariance $\underline{P}_o$ represents the Gaussian distribution about $\hat{\underline{x}}_o$, or the initial state uncertainty. Since $\underline{P}_o$ is not known a priori, an initial value must be assumed based on model parameters. Unfortunately, model parameters vary significantly for different types of engagements (i.e., a beam on or tail chase attack). Therefore, $\underline{P}_o$ is eventually selected, based on a sensitivity study (see Chapter V), such that the initial covariance is high. Then, through several filter propagation and update cycles the covariance converges to a reasonable value using the actual measurement history. In order to initially test a filter, a value of $\underline{P}_o$ must be assumed. Assuming symmetry in all axes (again not unreasonable), $\underline{P}_o$ is selected based on root-mean-square values of 25 feet for position, 200 feet/second for velocity, and three g's (96.6 feet/second) for acceleration (17). Thus,

$$\underline{P}_o = \begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 9300 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 9300 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9300 \end{bmatrix} \qquad (3\text{-}43)$$

3.3 Equivalent Discrete-Time EKF Design

The design presented in Section 3.2 is based on a continuous time dynamics model and is useful for evaluating filter performance. But for actual implementation, a continuous time dynamics model is not feasible because of the F-4E/G on-board digital computer. This motivates the need for an equivalent discrete-time system model. The discrete-time update relations remain identical to those presented in Section 3.2. Additionally, the equivalent discrete-time propagation relations can be reduced in complexity to that of a linear Kalman filter. These two points significantly reduce design complexity, which in turn reduces the memory storage and processing time required for a digital simulation or implementation.

3.3.1 Equivalent Discrete-Time System Model Design

As previously discussed, the measurements are only available at discrete-time intervals. Since the update relations of Section 3.2, Equations (3-5), (3-6), and (3-7), are based on discrete-time measurements, the update relations remain unchanged. The equivalent discrete-time propagation equations analogous to the continuous time equations of Section 3.2, Equations (3-8) and (3-9), are presented without derivation, using pages 170-172 of Reference 5 and pages 45-46 of Reference 16 as a guide. For the general case, the time propagations can be written equivalently as

$$\underline{\hat{x}}(t_{i+1}^{-}) = \underline{\hat{x}}(t_i^{+}) + \int_{t_i}^{t_{i+1}} \underline{f}[\underline{\hat{x}}(t/t_i),\underline{u}(t),t]dt \qquad (3\text{-}44)$$

$$\underline{P}(t_{i+1}^{-}) = \underline{\Phi}[t_{i+1},t_i;\underline{\hat{x}}(\mathcal{T}/t_i)]\underline{P}(t_i^{+})\underline{\Phi}^T[t_{i+1},t_i;\underline{\hat{x}}(\mathcal{T}/t_i)]$$
$$+\int_{t_i}^{t_{i+1}}\underline{\Phi}[t_{i+1},t;\underline{\hat{x}}(\mathcal{T}/t_i)]\underline{G}(t)\underline{Q}(t)\underline{G}^T(t)\underline{\Phi}^T[t_{i+1},t;\underline{\hat{x}}(\mathcal{T}/t_i)]dt \qquad (3\text{-}45)$$

Since $\underline{F}$ is linear and time-invariant, the stochastic difference equation reduces to standard linear propagation

$$\underline{x}(t_{i+1}) = \underline{\Phi}(t_{i+1},t_i)\underline{x}(t_i)+\underline{B}_d(t_i)\underline{u}(t_i)+\underline{w}_d(t_i) \qquad (3\text{-}46)$$

where,

$\underline{\Phi}(t_{i+1},t_i)$ is the state transition matrix and is derived in Equations (2-12) through (2-14),

$\underline{B}_d(t_i)$ is the discrete-time input matrix defined by

$$\underline{B}_d(t_i)=\int_{t_i}^{t_{i+1}}\underline{\Phi}(t_{i+1},\tau)\underline{B}(\tau)d\tau, \qquad (3\text{-}46a)$$

(assuming $\underline{u}(t_i)$ is constant on $t_i \leq t \leq t_{i+1}$) and $\underline{w}_d(t_i)$ is an s-vector-valued white Gaussian discrete-time stochastic process with mean zero and covariance kernel

$$E[\underline{w}_d(t_i)\underline{w}_d^T(t_j)] = \begin{cases} \underline{Q}_d(t_i) & t_i=t_j \\ \underline{0} & t_i \neq t_j \end{cases}$$

$$\underline{Q}_d(t_i)= \int_{t_i}^{t_{i+1}} \underline{\Phi}[t_{i+1},\tau]\underline{G}(\tau)\underline{Q}(\tau)\underline{G}^T(\tau)\underline{\Phi}^T[t_{i+1},\tau]d\tau \qquad (3\text{-}46b)$$

The time propagation relation for Equations (3-44) and (3-45) reduce to (5:275)

$$\hat{\underline{x}}(t_{i+1}^-) = \underline{\Phi}(t_{i+1},t_i)\hat{\underline{x}}(t_i^+)+\underline{B}_d(t_i)\underline{u}(t_i) \qquad (3\text{-}47)$$

$$\underline{P}(t_{i+1}^-) = \underline{\Phi}(t_{i+1},t_i)\underline{P}(t_i^+)\underline{\Phi}^T(t_{i+1},t_i) + \underline{Q}_d(t_i) \qquad (3\text{-}48)$$

3.3.2 Evaluation of Equivalent Discrete-Time Variables

To complete the design of the equivalent discrete-time model, $\underline{\Phi}(t_{i+1},t_i)$, $\hat{\underline{x}}_o$, $\underline{P}_o$, $\underline{B}_d$, and $\underline{Q}_d$ must be derived. Φ(ti+1,ti) is derived in Equations (2-11) through (2-14). Initial conditions $\hat{\underline{x}}_o$ and $\underline{P}_o$ remain as presented in Section 3.2.4.

For $\underline{B}_d$, evaluating Equation (3-46a) results in

$$\underline{B}_d = \left[\begin{array}{ccc} -\Delta t & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & -\Delta t & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & 0 & -\Delta t \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right] \qquad (3\text{-}49)$$

Evaluating Equation (3-46b) results in

$$\underline{Q}_d = \left[\begin{array}{ccccccccc} Q_{11} & Q_{12} & Q_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\ Q_{21} & Q_{22} & Q_{23} & 0 & 0 & 0 & 0 & 0 & 0 \\ Q_{31} & Q_{32} & Q_{33} & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & Q_{44} & Q_{45} & Q_{46} & 0 & 0 & 0 \\ 0 & 0 & 0 & Q_{54} & Q_{55} & Q_{56} & 0 & 0 & 0 \\ 0 & 0 & 0 & Q_{64} & Q_{65} & Q_{66} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & Q_{77} & Q_{78} & Q_{79} \\ 0 & 0 & 0 & 0 & 0 & 0 & Q_{87} & Q_{88} & Q_{89} \\ 0 & 0 & 0 & 0 & 0 & 0 & Q_{97} & Q_{98} & Q_{99} \end{array}\right] \qquad (3\text{-}50)$$

where, for the first block of the diagonal,

$$Q_{11}=[Q_1\tau_i^{5}/2][1-e^{-(2\Delta t/\tau_i)}+(2\Delta t/\tau_i)+(2\Delta t^3/3\tau_i^3)-(2\Delta t^2/\tau_i^2)-$$

$$(4/\tau_i)\Delta t e^{-(\Delta t/\tau_i)}],$$

$$Q_{12}=[Q_1\mathcal{T}_i^{\,4}/2][1+e^{-(2\Delta t/\mathcal{T}_i)}-2e^{-(\Delta t/\mathcal{T}_i)}+(2/\mathcal{T}_i)\Delta te^{-(\Delta t/\mathcal{T}_i)}-(2\Delta t/\mathcal{T}_i)+\Delta t^2/\mathcal{T}_i^{\,2}],$$

$$Q_{13}=[Q_1\mathcal{T}_i^{\,3}/2][1-e^{-(2\Delta t/\mathcal{T}_i)}-(2/\mathcal{T}_i)\Delta te^{-(\Delta t/\mathcal{T}_i)}],$$

$$Q_{21}=Q_{12},$$

$$Q_{22}=[Q_1\mathcal{T}_i^{\,3}/2][4e^{-(\Delta t/\mathcal{T}_i)}-3-e^{-(2\Delta t/\mathcal{T}_i)}+2\Delta t/\mathcal{T}_i\ ]$$

$$Q_{23}=[Q_1\mathcal{T}_i^{\,2}/2][1+e^{-(2\Delta t/\mathcal{T}_i)}-2e^{-(\Delta t/\mathcal{T}_i)}],$$

$$Q_{31}=Q_{13},$$

$$Q_{32}=Q_{23},$$

$$Q_{33}=[Q_1\mathcal{T}_i/2][1-e^{-(2\Delta t/\mathcal{T}_i)}],$$

$\Delta t=.04$ seconds (initially),

$\mathcal{T}i=0.5$ seconds (initially), and

Q1=Q2=Q3=37,325 initially (before tuning).

The second and third blocks of the diagonal are the same as the first but with the subscripts of Q_{xy} changed and Q_1 changed to either Q_2 or Q_3.

IV. METHODS OF MODEL SIMULATION AND TESTING

4.1 Introduction

The continuous time models contained in Chapter II and the Kalman filter equations in Chapter III are developed for simulation and testing utilizing a well established computer-aided design package called Simulation for Optimal Filter Evaluation (SOFE)(18). The equivalent discrete-time dynamics model and Kalman filter equations in Chapter III are developed for simulation on a digital computer as a stand-alone program for possible implementation on the F-4E/G The stand-alone simulation (SAS) program developed for this thesis is fully functional and provides performance similar to that of the SOFE simulation. The advantage of the SAS program over SOFE is that it is an equivalent discrete-time design containing the specific target estimation filter designed in Chapters II and III. Additionally, since it is an equivalent discrete-time design, it can easily be implemented on the F-4E/G digital computer. As such, the SAS program is recommended for further testing and possible implementation on the F-4E/G testing facilities or aircraft.

To demonstrate the validity of the models developed to this point, a computer program is used to generate truth model trajectory and measurement data for both the attacker aircraft and target aircraft. Then, the same trajectory data can be used to run either the SOFE or the SAS programs to evaluate the extended Kalman filter performance. The purpose of this chapter is to describe the truth model trajectory generation, SOFE implementation and testing procedures, SAS implementation and testing procedures, and the filter tuning philosophy employed.

4.2 Trajectory Generation (Truth Model)

An F-4E/G trajectory generation simulation model which provides the basic aircraft and target dynamics, problem geometry, radar model with antenna dynamics, and truth/measurement data required to perform a Monte Carlo analysis, was obtained from OO-ALC/MMECB (12). As discussed below, the trajectory simulation coding and output format is slightly modified in this thesis. The changes are necessary for computer system compatiblity and coordinate frame rotations into the filter frame. The computer simulation source code is included in Appendix B. The changes made are as follows:

1. The program received from OO-ALC included some non-standard (VAX VMS-specific) Fortran code. The non-standard coding is replaced by Fortran 5 coding so a standard Fortran 5 compiler can be used. In particular, for this effort, the code is compiled on a CDC CYBER compatible compiler which does not recognize the VAX VMS-specific code. The non-standard code (see Appendix B) is left in the source code as a comment line and the appropriate Fortran 5 statement entered after the comment line.

2. Target acceleration is changed from a horizontal acceleration to target accelerations in the target body axes. This is necessary to obtain the three-dimensional acceleration for use in the performance evaluation comparison of the acceleration truth states and filter states.

3. All the data required for SOFE or the SAS program, is rotated into one reference frame, the antenna reference frame (i_o, j_o, k_o). Filter states are also expressed in this frame, which allows direct comparison between the filter states and truth states.

4. Finally, by describing the filter states in a body reference frame (the antenna reference frame), the truth trajectory generation model is transformed from a two-dimensional simulation to a three-dimensional simulation. The trajectory generation program provided by OO-ALC is a two-dimensional simulation. The position, velocity, and acceleration states corresponding to the down direction from the horizontal plane are zero throughout the simulation. In order to obtain peformance evaluation in the down direction, the truth model is transformed into a three-dimensional simulation by an appropriate choice of fighter coordinate axes (thus avoiding a major truth model modification). The transformation occurs whenever the fighter rolls in the horizontal simulation plane. The states roll with the fighter, resulting in nonzero

values for states even though the simulation trajectory stays in the horizontal plane. In this manner, all modes and states of the three-dimensional problem are excited and therefore can be analyzed in the performance analysis.

The last three modifications described above are added to the basic trajectory generation as an user-selected option. Overall, the basic truth model received from OO-ALC is left intact. In fact, OO-ALC provided a sample trajectory output listing from the basic program. After all modifications were completed, the basic program option was rerun and the output trajectory found to be identical to that provided bv OO-ALC.

Using the modified trajectory generation program with the coordinate change option, two test trajectories are generated for Kalman filter testing. The trajectories used are a beam shot attack and a tail chase attack, both of which are described in detail below.

4.2.1 Trajectory Generation - Beam Attack

A beam attack trajectory simulation is selected for analysis since it is often employed in an air-to-air engagement. Additionally, the beam attack trajectory involves numerous fighter maneuvers which are of direct interest to this study (since the current Wiener-Hopf filter becomes unstable when the fighter maneuvers). The beam trajectory progresses as follows. Initially, the fighter and target are at a range of 40,000 feet, at the same altitude (arbitrary but below 32,000 feet, by definition from Section 1.4), both with an airspeed of 800 feet/second, and are flying at right angles to each other with the target 45 degrees to the right of the centerline extending from the fighter's nose (see Figure 4-1). One second after the simulation starts, the target starts a three-g level turn to its right. At three seconds, the fighter rolls right into the direction of the target and at after four seconds has established a two-g turn (approximately a 60-degree bank angle). This turn is maintained from simulation time of four to five seconds. At five seconds, the fighter starts a reverse roll to the left and rolls out essentially wings-level shortly after six seconds. A constant fighter heading is maintained for the rest of the simulation. At 9 seconds, the target starts a left roll and attains essentially straight-and-level flight at about 10 seconds. The simulation is terminated at 12 seconds.

4.2.2 Trajectory Simulation - Tail Chase

The tail chase trajectory simulation is selected for analysis for the same reasons as the beam attack trajectory. The tail chase trajectory progresses as follows (see Figure 4-2). Initially, the fighter and target are at a range of 10,000 feet with the same heading (arbitrary), same altitude (arbitrary but below 32,000 feet), and at the same airspeed (800 feet/second). The target is five degrees left of the fighter's nose. At one second, the target starts a

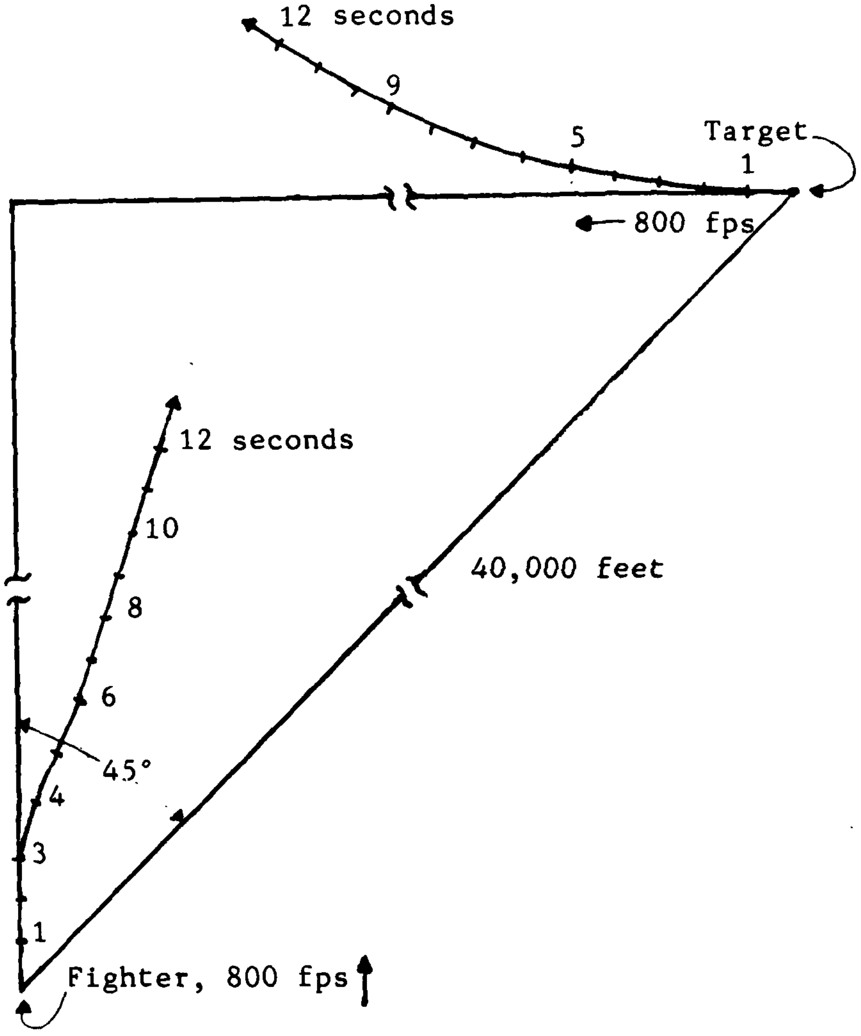


Figure 4-1. Beam Attack Trajectory - Top View (not to scale)

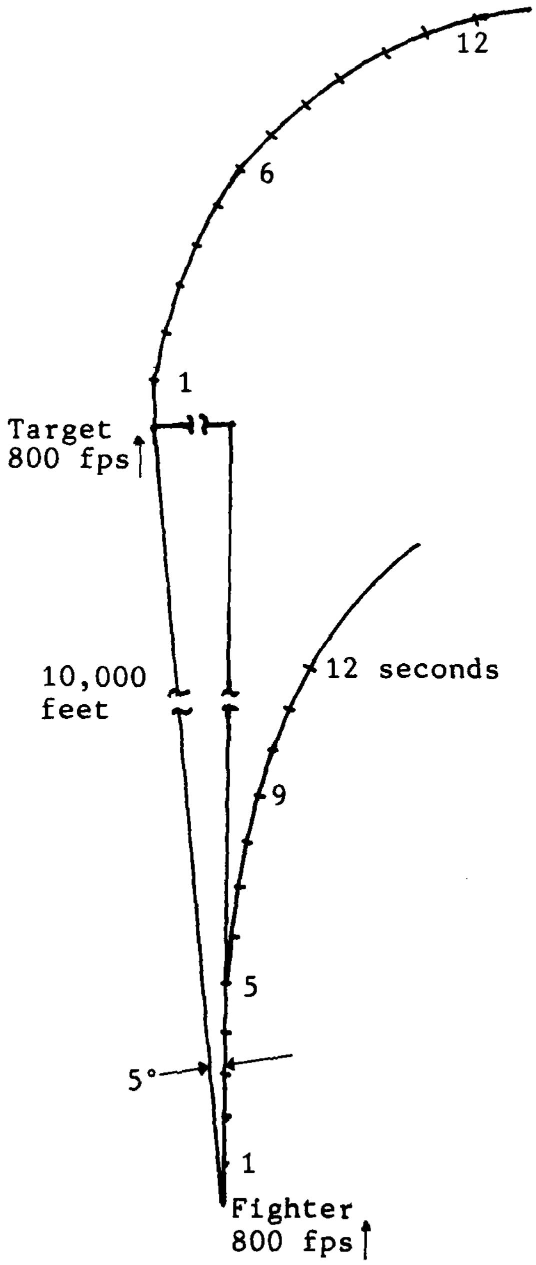


Figure 4-2. Tail Chase Trajectory - Top View (not to scale)

three-g level turn to the right, turning in front of the fighter. At 5 seconds, the fighter rolls right in pursuit of the target and then (at about 6 seconds) maintains a 2-g turn (approximately a 60-degree bank) which is held for the remainder of the simulation. The target crosses in front of the fighter's nose at about 5.5 seconds. Then, at about 6.6 seconds the fighter's nose catches up to the target and the target is once again slightly to the left of the fighter's nose. The fighter "tracks" the target for the remainder of the simulation with the target staying within one degree of the fighter's nose.

4.2.3 Trajectory Simulation Run Time Selection

The trajectory simulations are run at 0.02 second intervals and the results are stored in an external data file. SOFE input routines read and interpret trajectory and measurement data from the stored data file, while the SAS program must have an update time that is an even multiple of, or equal to, the trajectory simulation time. Since some of the simulation completed in this effort is run at a 0.04 second update period (as explained in Section 4.3.2), 0.02 is selected for the trajectory generation simulation time to allow points on either side of the SOFE interpretation. This in turn, allows a SAS update period of 0.1 seconds.

4.3 SOFE Simulation and Testing - System Validation

4.3.1 Introduction

The purpose of the SOFE simulation is to evaluate how the sampled-data continuous-time Kalman filter performs compared to the truth model (trajectory generation of last section). SOFE and a SOFE Plotting (SOFEPL) (19) program are used as basic tools for generating the data required for filter evaluation. These programs were developed by the Air Force Avionics Laboratory as general-purpose programs to help design and evaluate Kalman filters for integrated systems. SOFE provides the basic functions required to perform a Monte Carlo analysis on the extended Kalman filter designed in earlier chapters. SOFE contains 31 routines to perform such tasks as input/output, problem and run setup, numerical solution to ordinary differential equations, measurement update through a Carlson square root algorithm (5:385), and run and problem termination (18:Abstract). Nine user-written subroutines define the specific system and extended Kalman filter under study. The user-written routines used in this thesis are included in Appendix C. Equations (2-9), (2-10), (2-33), (2-34), (3-15) through (3-39), (3-42), (3-43), and (3-50) are incorporated into the user subroutines to specify an integrated Kalman filter design. The outputs of SOFE are stored for postprocessing by SOFEPL. SOFE outputs records for each time increment containing time, the truth states, the diagonal elements of the filter computed covariance, the measurement residuals, and the residual variances. The output records are postprocessed by SOFEPL to form an ensemble of Monte Carlo runs and DISSPLA (20) to produce graphical Calcomp (21) plots of the Monte Carlo analysis.

Figure 4-3 illustrates a sample Calcomp plot generated during this research. The plot first displays the mean of the error states (state estimate minus the truth state). Second, the envelope formed by the mean error plus or minus the standard deviation of the actual error is illustrated. Finally, the envelope formed by zero plus or minus the square root of the sampled-averaged filter-computed covariance diagonal terms is displayed.

4.3.2 Determining Filter Performance

The plots generated from SOFE/SOFEPL/DISSPLA/Calcomp (SSDC) processing are used to determine how a particular Kalman filter performs. Performance is determined by comparing different plots for particular tuning factors. Ideally, the true mean error plus or minus the one-sigma time histories should stay inside the envelope formed by zero plus or minus the square root of the corresponding filter-computed covariance diagonal terms (see Figure 4-3). For a properly tuned filter, shortly after a target maneuver, the mean error data may stray outside the filter-computed standard deviation envelope but should return. Plots from various simulation runs having different tuning values of $\underline{R}$ (see Equation (2-34), $\underline{Q}$ (see Equation (3-42), and tau (see Equations (2-9) and (2-10)) are compared to determine which set of tuning

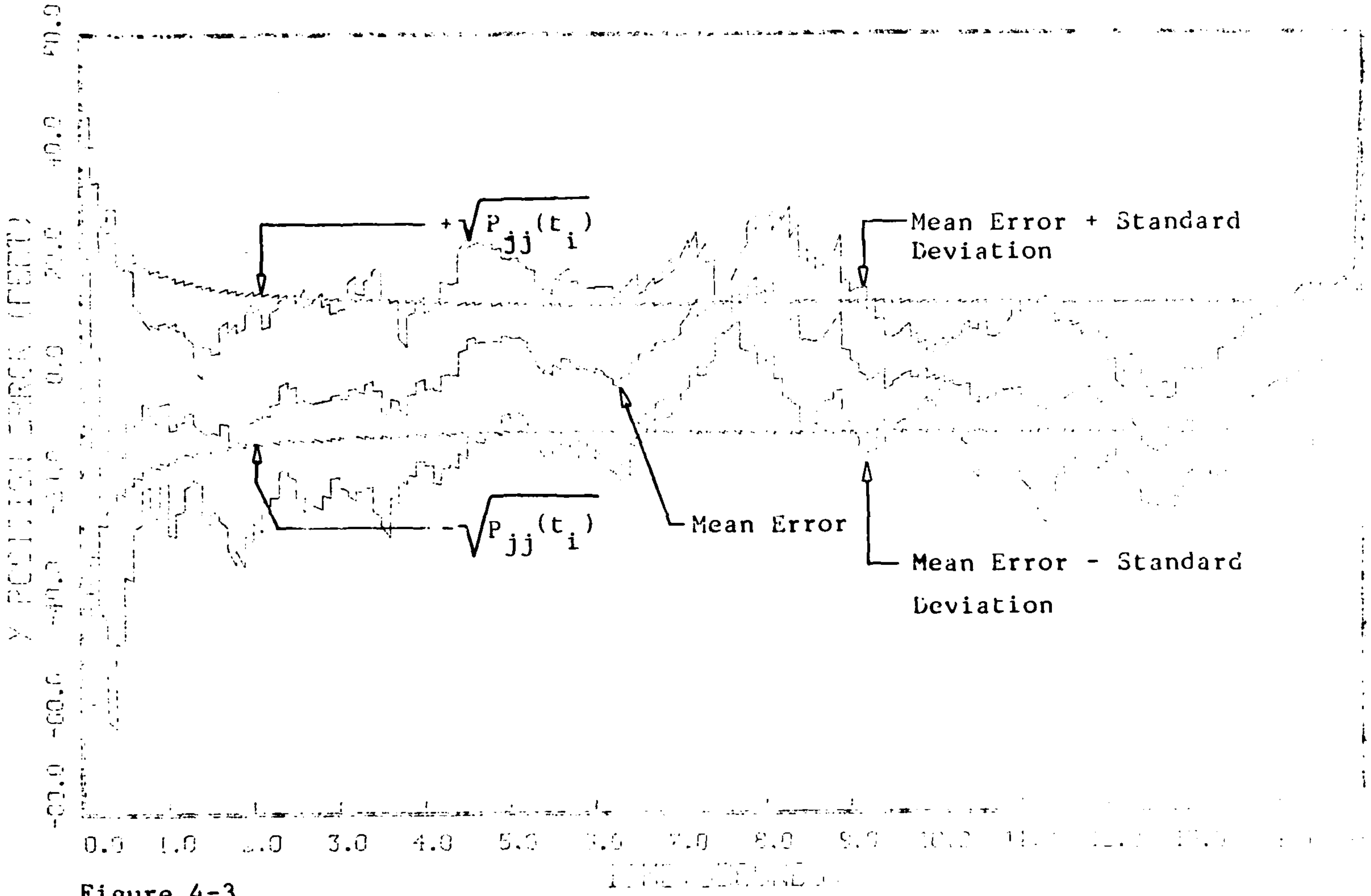


Figure 4-3

STATE 1, Q(1)=Q(2)=Q(3)=11300., TAU(1) .15,TAU(2-3)=.1[illegible]

[illegible] TAIL CHASE, INITIAL [illegible] [illegible] 0.[illegible], U-DATE=.1 [illegible]

Representative Output from SSDC Processing

factors performs the best, i.e., which yields the least mean error and results in the true mean plus or minus the standard deviation staying reasonably within the square root of the filter-computed covariance diagonal elements.

Figure 4-4 presents one plot of sample statistics computed from five runs of the overall problem. A velocity state mean error plot is chosen since the velocity states consistently demonstrate the most sensitivity to different tuning factors and update periods. For this particular problem, five runs are sufficient for evaluation between different tuning factors. This can most easily be demonstrated by comparing Figures 4-4 and 4-5. Figure 4-4 is a five-run analysis and Figure 4-5 is a 20-run analysis, both with an update time of 0.04 seconds. Note that both figures display the same general shapes and magnitude. Thus a five-run analysis essentially provides the same information as a 20-run analysis but at significantly reduced computer cost. To conserve further on computer costs, the update period is increased from 0.04 to 0.1 seconds. Again, it is easiest to justify this by comparing Figures 4-4 and 4-6. Note that both figures have similar shapes and magnitudes. Therefore, the trends of the tuning parameters $\underline{R}$, $\underline{Q}$, and tau can be observed from a five-run analysis with an update of 0.1 seconds as effectively as a 20-run, 0.04 update Monte Carlo anlysis. Initially, it was thought, once the final

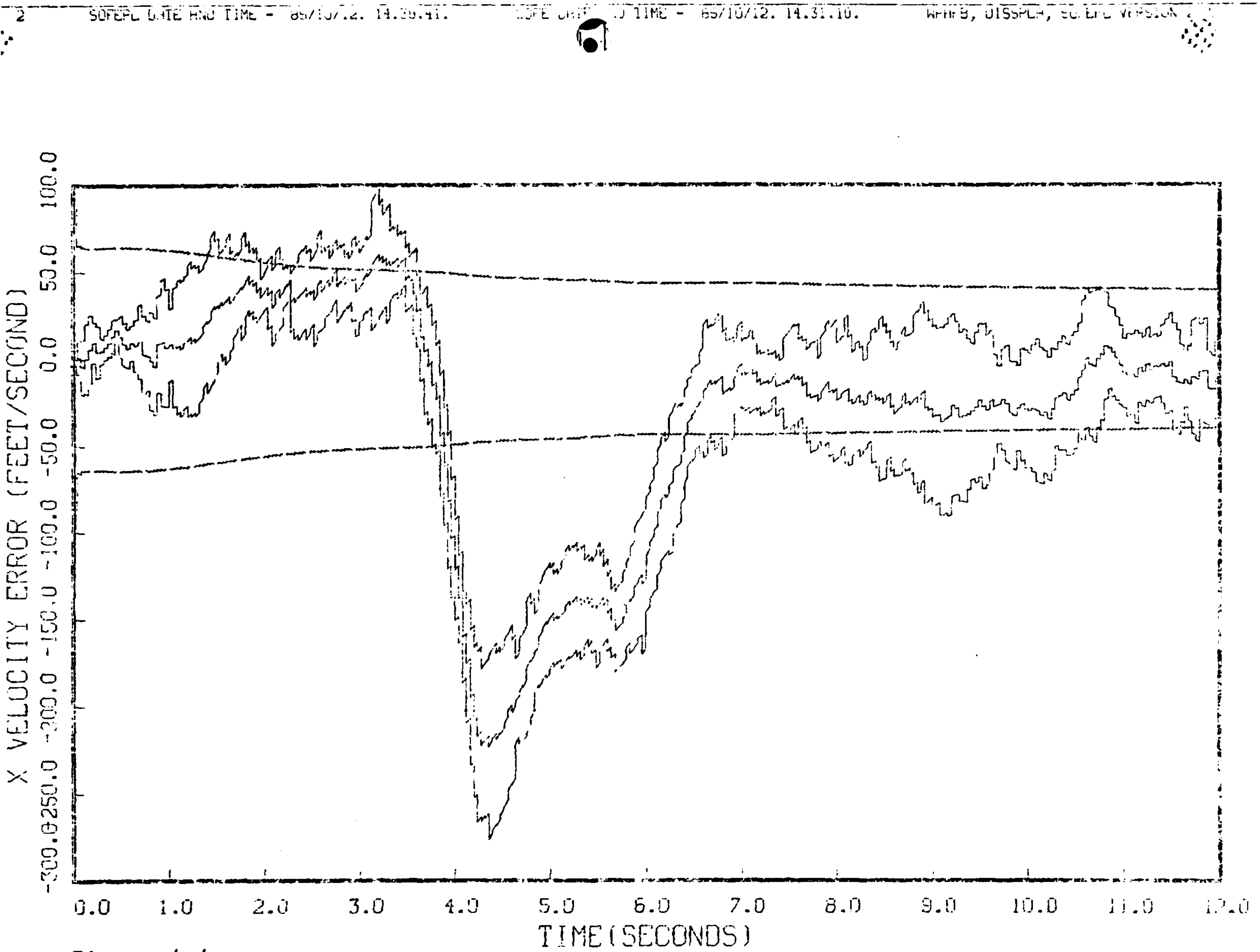


Figure 4-4

STATE 2, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

Filter Performance Example

3 SUPERC DATE AND TIME - [illegible] SUPE [illegible] AND TIME - 85/10/03. 22.14.32. [illegible] SUPERC VERSION

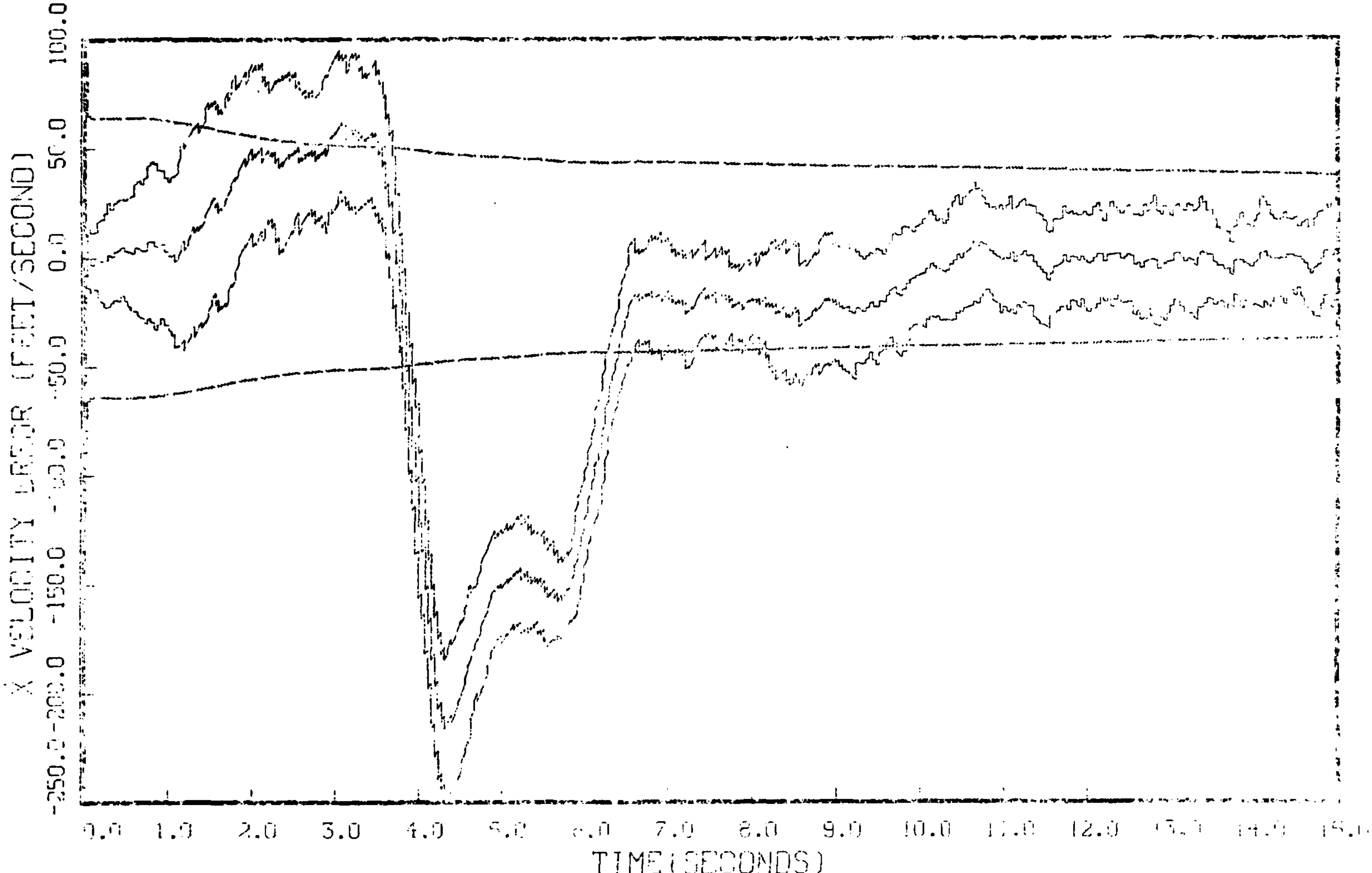


Figure 4-5

STATE 2, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS APG-120, BEAM ATTACK, INITIAL RANGE=10,000.', UPDATE=.01, 30 RUNS

Filter Performance Example

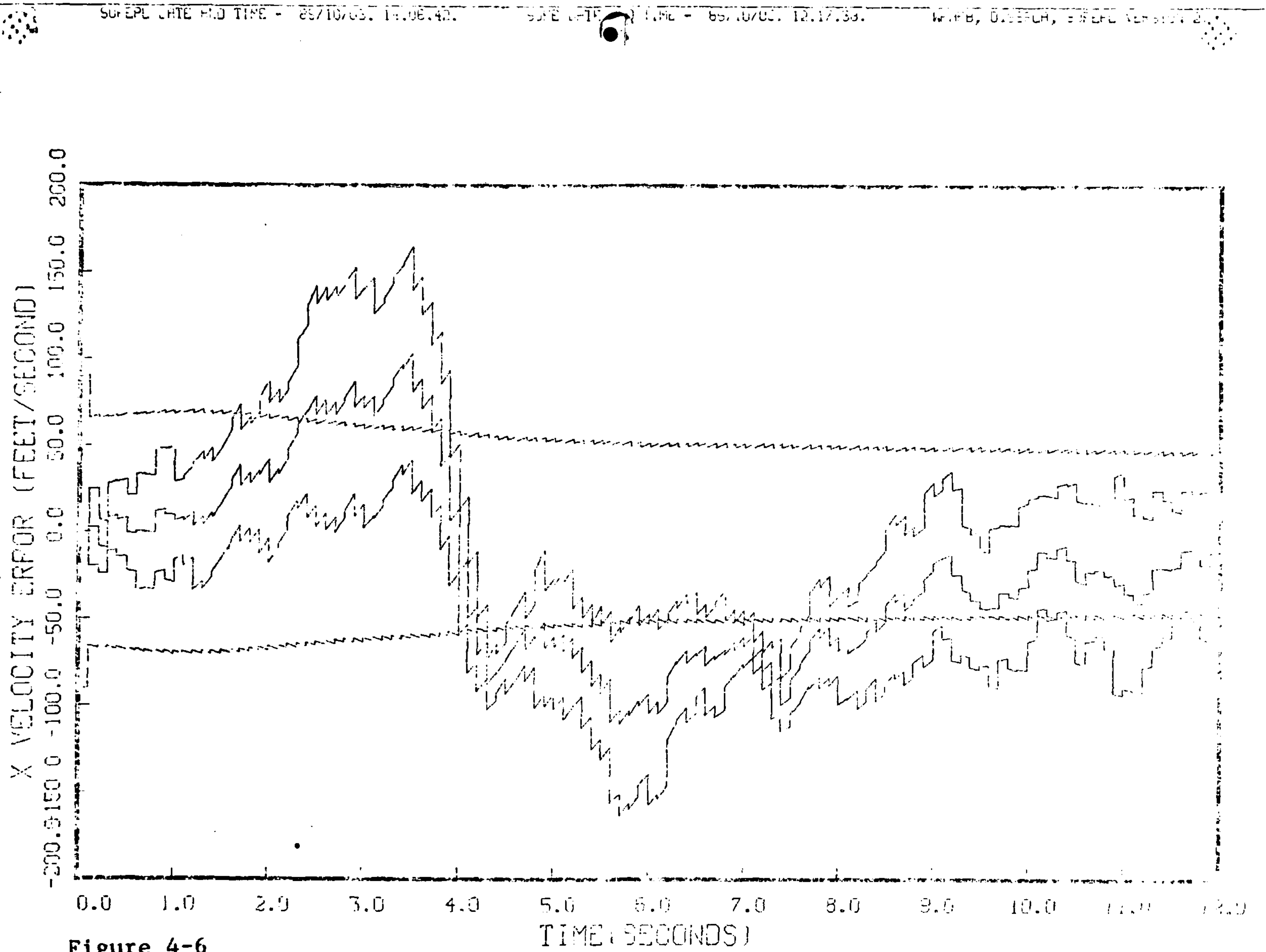


Figure 4-6

STATE 2, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

Filter Performance Example

tuning parameters were selected based Figure 4-4 on a five-run analysis with update time of 0.1 seconds, the number of runs and update time could be reset to 20 and 0.04 seconds respectively to verify performance. In this manner, only a relatively small number of large Monte Carlo runs need be completed. However, as shown in Figures 4-4 and 4-6, increasing the update period to 0.1 seconds causes a change in the velocity mean error. This implies the update period is actually a tuning parameter, similar to $\underline{R}$, $\underline{Q}$, and tau. To illustrate this, the same simulation is rerun at an update period of 0.1 seconds (see Figure 4.7) but with $\underline{Q}$ arbitrarily decreased by a factor of 2.5 (since the update period is increased by a factor of 2.5). Figure 4-7 indicates performance closer to Figure 4-4 than Figure 4-6. Thus, as determined from observed performance, the filter will have to be fine tuned whenever the update period is changed. Analytically, as shown in Equation (3-50), $\underline{Q}_d$ is a function of the update period, Δt. Thus the update period is actually a tuning parameter and should be considered during follow-on aircraft implementation. But, for the purpose of this study, the majority of the simulations are completed at an 0.1 update period to conserve on computer costs. The difference between the various figures are detailed numerically in subsections under Section 5.3.1.

2 SOFEFL DATE AND TIME - 85/10/15. 20.09.31. SOFE DATE AND TIME - 85/10/16. 21.57.00. WFHFB, DISSPLR, SOFEFL VERSION

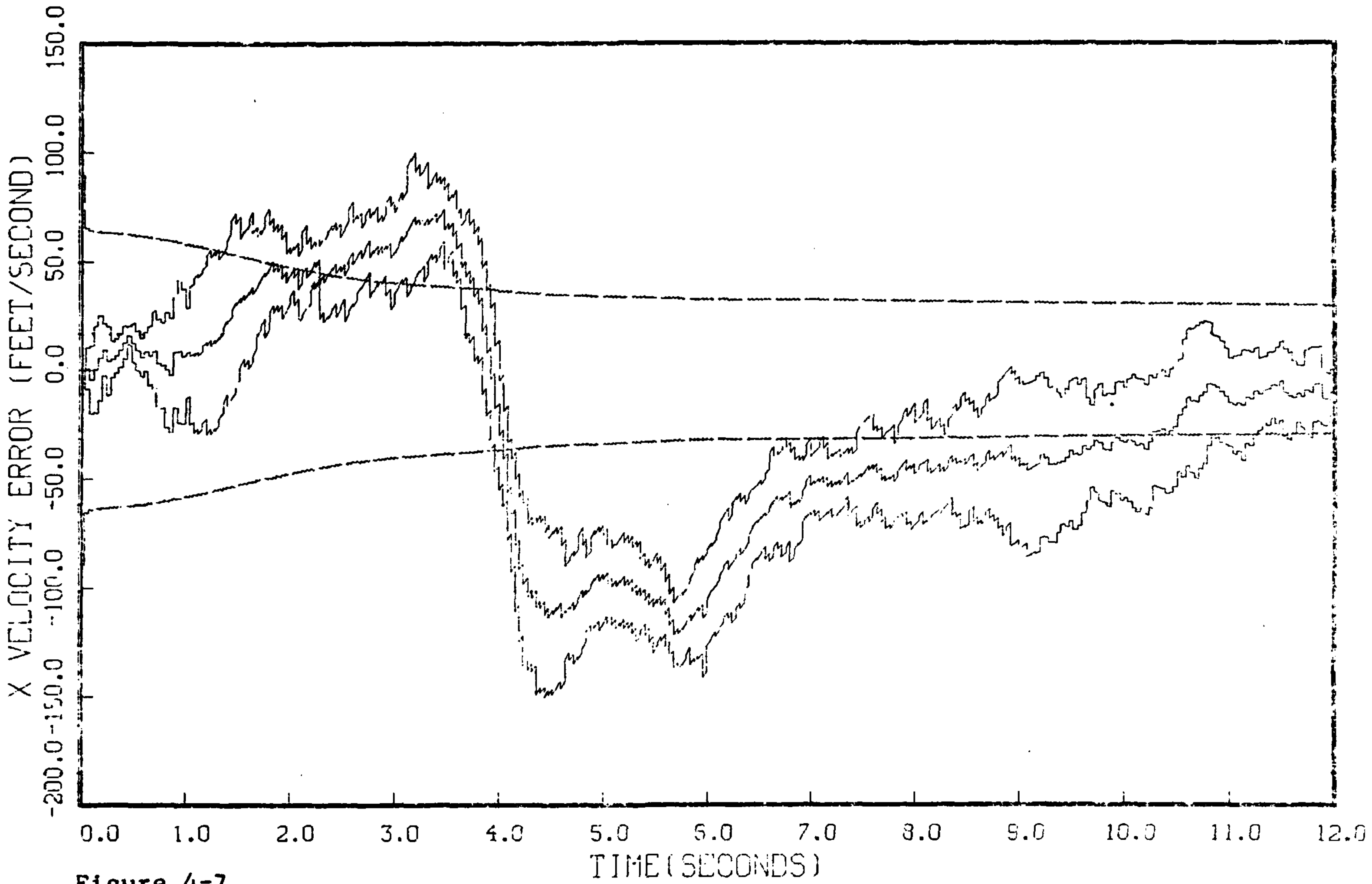


Figure 4-7

STATE 2, Q(1)=Q(2)=Q(3)=59720., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APC-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=0.04, 5 RUNS

Filter Performance Example

4.3.3 SOFE Modification

In order to use SOFE for this particular problem, a modification is made to the basic SOFE program. This is necessary because the dynamic model for this research is referenced to the antenna reference frame (i_o, j_o, k_o), which is a body frame, not an inertial frame. The Kalman filter propagation from time t_{i-1}^+ to t_i^- is accomplished in an inertial type frame (based on the last t_{i-1}^+ estimates), and then $\hat{\underline{x}}(t_i^-)$ and $\underline{P}(t_i^-)$ are impulsively rotated to the body frame axis at time t_i. SOFE is modified to perform the required rotations. Mathematically, both the terms in the right-hand side of Equations (3-5) and (3-6) must be in the same frame. At time t_{i-1}^+ the state reference is propagated to time t_i^-. If the state reference is redefined because of any change in heading, pitch, or roll during the propagation cycle, the required information at time t_i^- must be rotated into the new reference frame at time t_i in order to apply the update relations properly. This modification, along with subroutine calls, are included in Appendix D.

4.4 Stand-Alone Simulation (SAS) - Simulation and Testing

4.4.1 Introduction

The purpose of the equivalent discrete-time models and Kalman filter development in previous chapters is to determine how well a discrete-time filter performs. Additionally, as previously discussed in Section 3.3, the filter implementation on the F-4E/G will be an equivalent discrete-time design. Thus, the SAS program permits analysis of a discrete-time filter while also containing the basic code required for implementation. The SAS program developed in this thesis is included in Appendix E.

4.4.2 Equivalent Discrete-Time Algorithm Design Considerations

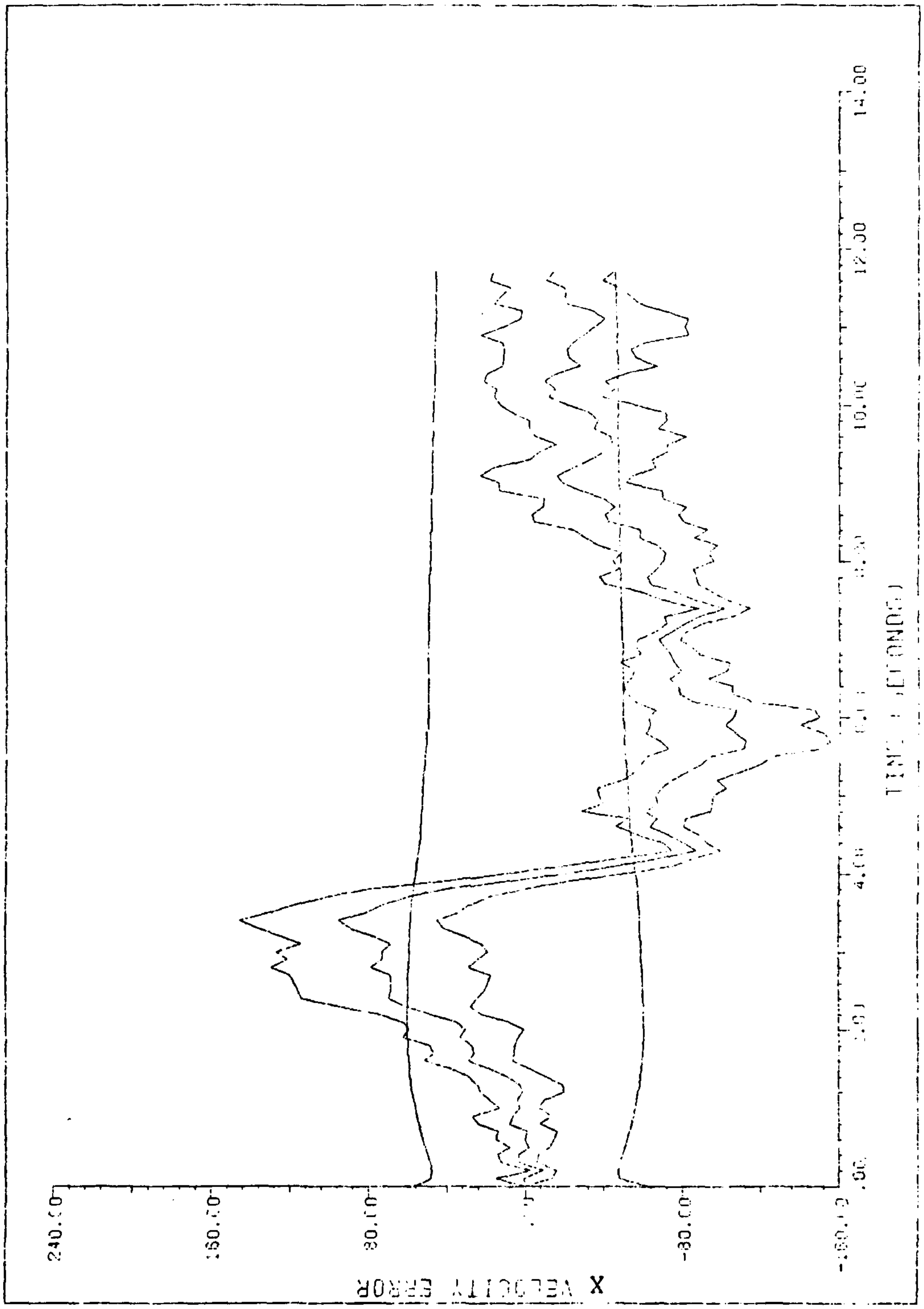
In simplistic terms, propagation equations (Equations (3-48) and (3-49)) and update equations (Equations (3-4), (3-5), and (3-5)) are implemented. Due to limited memory storage (4k or possibly 8k words), desired update period (0.04 to 0.1 seconds), and limited arithmetic capability of the F-4E/G fire control computer, a conventional Kalman filter algorithm utilizing matrix operation routines is selected for the preliminary design. Matrix routines are initially used in the SAS program for simplicity, but if the nine-state filter designed is implemented on the F-4E/G, the matrix routines should be simplified by taking advantage of the number of zero elements and matrix symmetry. Additionally, the SAS measurement update algorithm should be modified into a U-D filter form to increase numerical stability (5:391-396). Another reason for using a U-D filter is to avoid as many divides and square roots as possible. A divide function on the F-4E/G computer consumes 24.33 μsec while a square root consumes 150.0 μsec, which can rapidly use up the available 40 to 100 msec processing time available. As a comparison, other commonly used mathematical functions such as addition, subtraction, and multiplication consume 9.0 μsec or less. Appendix F contains calculations which show the U-D filter satisfies the system limitations defined in Section 1.3. Alternate forms of update algorithms such as the Potter covariance square root, Carlson square root, and inverse covariance, probably should not be considered because of either the required number of divides or square roots.

4.4.3 Determining Filter Performance

SAS filter performance and filters simulated using SOFE are evaluated in the same manner (see Section 4.3.2). The SAS program stores the required data to generate plots through a postprocessor similar to those generated in the SSDC processs. Again, the easiest way to demonstrate performance is to examine plotted output. Figures 4-6 and 4-8 illustrate the simulated performance for the filter using SSDC and the SAS process respectively (for the same beam attack trajectory). Note the performance is essentially the same; the performance is compared numerically in subsections of Section 5.3.1.

4.5 Filter Tuning Philosophy and Methods

Filter tuning is required to achieve the best filter performance, compensating for the modeling approximations made during reduced order filter design. As previously noted, four factors can be varied to affect filter tuning. These are the $\underline{R}$ matrix, the $\underline{Q}$ matrix, the value of tau used in the acceleration filter states, and the update period. In



AFO120 Q=148300, TAU=.148, 5 RUNS. 6 MEAS

Figure 4-8 SAS Generated Plot for Comparison to SSDC Process

this thesis, $\underline{R}$ (see Equation (2-34)) is treated as a constant (after initial testing and verification), lumping together system and radar antenna lag noise. Then, $\underline{Q}$ (see Equation (3-42)) and values of tau (see Equation (2-10)) are varied to achieve the best tuned performance (some comparison is made for the effect of the update period).

For this particular problem, good estimates of target velocity are desired more then precision in estimation of position and acceleration states (24). Thus, the philosophy employed is to achieve a minimum velocity error while maintaining errors in position and acceleration less than the corresponding values from the Wiener-Hopf filter.

V. SIMULATION RESULTS

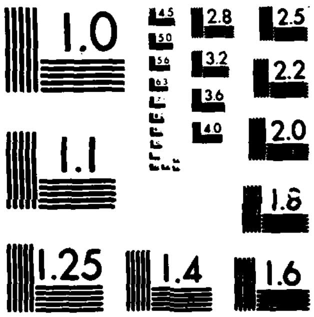
5.1 Introduction

The development to this point is based on determining the feasibility of replacing the existing F-4E/G Wiener-Hopf target estimation filter with a Kalman filter. To accomplish this, a preliminary nine-state Kalman filter is designed and tested through simulation for comparison to the Wiener-Hopf filter. This chapter includes the results of the computer simulations performed on the preliminary design. As will be shown, the results indicate that the preliminary design significantly outperforms the Wiener-Hopf filter for the beam shot attack. Data for the Wiener-Hopf filter performance on the tail chase is not available and is not directly compared to the Kalman filter results. However, even with increased performance on the beam attack, further testing/remodeling is desirable to provide additional performance enhancement. Overall, it can be said this study demonstrates the Kalman filter is a feasible choice for further study/testing to replace the Wiener-Hopf filter eventually.

5.2 Preliminary Kalman Filter Design

The preliminary design Kalman filter contains nine states and uses six measurements for the update. The states are described in Section 2.4.3 and the measurements in Section 2.5.2. Linear dynamics are employed for filter propagation and nonlinear measurements are used for the filter update, as described in Section 2.2.

AD-A164 034 PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT 2/5
MANEUVERING TARGET..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R B ANDERSON
UNCLASSIFIED DEC 85 AFIT/GE/ENG/85D-2 F/G 19/5 NL


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

5.3 Computer Simulation Results

The computer simulation methodology is discussed in Chapter IV. Four different simulation groups are summarized in this chapter. First, the beam trajectory is tested extensively since it provides the most challenge, as it contains the largest errors in velocity estimates. Additionally, a direct comparison to the Wiener-Hopf filter performance is possible. Second, the tail chase is briefly studied to demonstrate how the tuning values derived for the beam attack trajectory perform in the tail chase scenario. Third, "off-line" (see Section 3.1) adaptive tuning is tested to show that further study of on-line adaptive estimation is warranted and desirable. Finally, it is determined that overall Kalman filter performance, while superior to the Wiener-Hopf filter, is less than desired. Therefore, the last simulation group attempts to isolate the major source of performance degradation and to provide insight where future study should start.

5.3.1 Beam Attack Simulation

Plots illustrating Wiener-Hopf filter performance for the beam attack were provided by OO-ALC/MMECB (22) and are included in Appendix G. These plots are used as a baseline of performance from which enhanced performance is desired. Appendix G also includes nine plot sets from simulation runs of the same trajectory using the preliminary design Kalman filter. The plot set figure numbers correspond to the table numbers of this chapter. For example, Figure G.2.3.a is interpreted as

G: Appendix G

2: Plot Set 2 (also indicates data is in Table V-2)

3: Third column of tabulation in Table V-2

a: x position error (b: x velocity, c: x acceleration, d: y position, e: y velocity, f: y acceleration, g: z position, h: z velocity, and i: z acceleration errors) (Note, that x,y,z axes on the plot sets are i_o, j_o, k_o axes, not space coordinates.)

Additionally, units for all the tables are

feet for position states (x_1, x_4, and x_7),
feet/second for velocity states (x_2, x_5, and x_8),
feet/second^2 for acceleration states (x_3, x_6, and x_9),
feet^2/second^5 for $\underline{Q}$, and
seconds for tau.

SOFE/SOFEPL/DISSPLA/Calcomp (SSDC) simulation is primarily used for filter evaluation but the stand-alone simulation (SAS) program (see Section 4.4) is also verified. As discussed in Section 3.2.3, $\underline{R}$ is not varied once it is initially selected (through analysis in Section 2.5.2 and simulation verification). The first plot set is presented to show how a range of $\underline{Q}$ is initially selected. With a $\underline{Q}$ range selected, plot set two demonstrates how tau is selected. Plot set three then illustrates how a final value of $\underline{Q}$ is selected. With the values of $\underline{Q}$, $\underline{R}$, and tau selected, the SAS program results are included in plot set four to demonstrate the SAS program equivalence to the SOFE/SOFEPL/DISSPLA/ Calcomp process. Finally, plot set five compares a six-measurement tuned filter, the same filter with four measurements (dropping azimuth and elevation rate measurements), and the Wiener-Hopf filter baseline performance. Note that either the maximum, the minimum, the average, the standard deviation, or a combination thereof, of the mean error values from the plotted output of Appendix G are included in the tables. Other representations are possible, but the mean error is tabulated because of the minimum velocity error tuning policy established in Section 4.5. The results of the plot sets are tabulated in the following subsections.

5.3.1.1 Plot Set One - Selecting a Q Range

Plot set one is used to select an initial value Q range for tuning purposes. Following the tuning policy of Section 4.5, velocity errors are to be minimized instead of position or acceleration errors. Additionally, a reasonable settling time to essentially zero-mean error for all the states is desired. For the beam shot trajectory analyzed in this research, 12 seconds total elapsed time is selected for all states to return to approximately zero-mean error. This corresponds to a simulation time of six seconds after the fighter stops maneuvering and three seconds after the target stops maneuvering. Tau of 0.2 seconds is initially used based on previous simulation results (not included, but the effects of tau are demonstrated in plot set two). $\underline{P}_o$ is doubled from that calculated in Equation (3-43), based on previous simulation results (not included, but Appendix G filter-computed covariance curves are observed to be reasonable). Plot set one results are condensed in Table V-1.

Table V-1

Selection of $\underline{Q}$ Range
(Maximum Mean Error Magnitude / Elapsed Time to Settling)

	$\underline{Q}$ values (tau=0.2, update period 0.1 seconds)				
State	373.25	3732.5	37325.	373250	3732500
x_1	700/-	750/-	700/7.1	560/7.4	600/8.0
x_2	170/-	130/-	100/12	310/6.4	800/7.4
x_3	96/9.7	96/9.2	90/9.1	95/9.1	200/7.0
x_4	1600/-	1600/-	1500/7.0	1300/7.0	825/8.0
x_5	325/-	170/-	270/12	900/7.2	2500/7.2
x_6	60/9.4	60/9.4	60/9.3	100/9.1	500/6.8
x_7	2600/-	2400/-	2250/-	2150/+	1600/8.0
x_8	225/-	240/-	425/-	775/10.5	1550/8.2
x_9	30/6.7	30/6.7	30/6.7	30/9.0	30/9.0

- indicates state not settled at 12 seconds
+ indicates state almost settled at 12 seconds
Note: All data points read off plots

Heuristically, increasing $\underline{Q}$ weights the incoming measurements more than the internal filter propagation. In Table V-1, $\underline{Q}$ is varied by an order-of-magnitude or a factor of 10 to expedite the selection process (this is why the significant figures are carried, to clearly indicate the order-of-magnitude changes). Note that with the lower $\underline{Q}$, the mean velocity errors (states x_2, x_5, and x_8) are smaller but the mean position errors (states x_1, x_4, and x_7) are larger and these states are not settled by 12 seconds. Raising $\underline{Q}$ to 373250 increases the mean velocity errors but causes a desired reduction in both the mean position error and settling times. Raising $\underline{Q}$ to 3732500 results in large increases in velocity and acceleration errors (for example, velocity state x_5 increases by a factor of 2.5 and acceleration state x_6 increases by a factor of 5). Thus, results of Table V-1 indicate that the best velocity performance and reasonable settling time of about 12 seconds are obtained in a $\underline{Q}$ range of of 37325 to 373250. This corresponds to an order-of-magnitude change in $\underline{Q}$ and is in the range calculated in Equation (3-42).

5.3.1.2 Plot Set Two - Selection of Tau

A $\underline{Q}$ value of 373250 is selected out from the $\underline{Q}$ range established in the last section for fine tuning of tau. Again, following the tuning policy of Sections 4.5 and 5.3.1.2, mimimal velocity error and reasonable settling time are desired. The plotted results are condensed in Table V-2. Note that a value of tau equal to 0.5 seconds results in good settling time characteristics but at the expense of large mean velocity errors. Decreasing tau to 0.143 (which corresponds to a $1/T$ of 7.0 in Equation (2-10)) increases the settling time for some of the states but reduces velocity mean errors. Decreasing tau further results overall in increased settling times and approximately a 75 feet/second bias in state x_5 between 7 and 10 seconds. Results of Table V-2 indicate that a tau of 0.143 results in the best velocity performance with most of the states almost settled by 12 seconds.

Table V-2

Selection of Tau
(Maximum Mean Error Magnitude / Elapsed Time to Settling)

Tau Values (Q=373250, update period 0.1 second)

State	0.5	0.2	0.167	0.143	0.1
x_1	550/8.2	560/7.4	572/7.4	588/7.2	634/9.0
x_2	670/7.4	310/6.4	251/7.7	203/7.7	119/9.0
x_3	300/7.0	95/9.1	114/9.2	106/9.3	98/9.3
x_4	1000/8.2	1300/7.0	1333/7.0	1379/6.5	1465/7.5
x_5	1950/7.6	900/7.2	740/7.0	620/6.5	415/12.[1]
x_6	600/9.3	100/9.1	89/9.1	83/9.1	72/9.1
x_7	1750/8.1	2150/12+	2116/12+	2152/12+	2213/12+
x_8	1450/9.0	775/10.5	705/12.	640/12+	542/-
x_9	300/9.3	30/8.3	41/7.0	31/7.0	30/7.0

- indicates state not settled at 12 seconds
+ indicates state almost settled at 12 seconds
[1] indicates state is biased by 75 feet/second between 7 and 10 seconds
Note: data points read off plots

5.3.1.3 Plot Set Three - Selection of Q

With a tau of 0.143, Q is now fine tuned. The range of Q is divided into increments of 37325, 149300, 261275, and 373250 (calculated by subtracting 37325 from 373250, dividing by three resulting in 111,975.0, and adding this figure to 37325 successively to obtain the values listed). Again, the best velocity performance and all states settled by 12 seconds are desired. The results of simulation runs based on these values are contained in Table V-3. Data points are obtained from listable output of SOFEPL. Table V-3 indicates the best value of Q is 149300. Again, the plot sets must be studied for settling time characteristics. If the settling time associated with a particular Q is considered too long, then a higher Q can be selected to shorten the settling time. Additionally, as shown in plot set two, tau can also be varied to affect settling time, or as is later shown in plot set four, the update period can be decreased to 0.04 seconds. However, it is easily observed that decreasing the settling time increases the velocity error. The tradeoff here can be partially overcome by adaptive tuning as discussed in Section 3.1 and later demonstrated in Section 5.3.3. Finally, for Q equal to 149300, plots of the measurement residuals are included in Appendix G to provide insight on what the residuals are doing corresponding to the state estimate errors. Note that when the state errors are large, the angle and rate measurement true residuals statistics cross over

Table V-3

Selection of $\underline{Q}$ (Mean Errors)

State	$\underline{Q}$ value (Tau=0.143, update period 0.1 seconds)			
	37325	149300	261275	373250
x_1-minimum	-94.3	-82.6	-70.3	-57.1
-maximum	722.5	647.0	609.0	588.4
-average	105.2	94.4	91.5	90.7
-std dev	222.0	190.1	172.3	160.9
x_2-minimum	-79.4	-110.5	-153.1	-203.3
-maximum	111.2	103.3	101.3	99.7
-average	-10.9	-22.4	-25.1	-26.1
-std dev	50.3	51.9	56.4	61.7
x_3-minimum	-6.8	-20.6	-30.6	-38.3
-maximum	92.4	94.2	98.1	106.4
-average	45.8	44.8	43.2	42.2
-std dev	35.8	34.4	35.0	36.2
x_4-minimum	-1573.9	-1485.0	-1426.4	-1379.5
-maximum	405.0	514.4	569.9	604.3
-average	-58.1	-45.3	-39.5	-36.4
-std dev	362.4	346.1	337.1	330.9
x_5-minimum	-87.3	-96.3	-99.5	-101.4
-maximum	210.2	371.0	506.8	620.8
-average	63.4	71.5	75.5	78.0
-std dev	76.7	110.8	139.0	162.7
x_6-minimum	-8.4	-27.2	-37.8	-45.0
-maximum	65.4	65.4	75.2	83.1
-average	17.2	14.1	12.6	11.6
-std dev	20.1	20.3	22.2	24.7
x_7-minimum	-970.2	-966.6	-964.5	-962.0
-maximum	2308.9	2226.4	2168.8	2152.1
-average	336.2	235.2	208.4	196.3
-std dev	796.3	751.6	720.4	697.6
x_8-minimum	-361.9	-516.8	-590.0	-640.3
-maximum	81.7	100.3	127.0	150.7
-average	-119.9	-163.1	-173.1	-176.6
-std dev	161.9	206.5	227.5	243.5
x_9-minimum	-27.7	-28.8	-29.9	-31.3
-maximum	1.7	5.8	14.1	22.3
-average	-3.5	-2.2	-1.4	0.7
-std dev	7.4	7.2	8.4	10.0

the filter computed one-sigma curves. The large errors are attributed to a combination of improper filter tuning and modeling. The residual information can be used to either "retune" the on-line filter through adaptive estimation techniques (16:Chapter 10) or to recalculate state estimates through ad hoc techniques (as suggested later in Section 6.2.3 for future study).

5.3.1.4 Plot Set Four - Simulation Equivalency

Plot set four demonstrates the degree of equivalence between various simulation runs completed. First, the SAS results (five-run analysis) are compared to the SSDC results (also a five-run analysis). Second, the SSDC for the five run analysis at an update period of 0.1 seconds is compared to an equivalent twenty-run analysis. Next, a SSDC analysis at an update period of 0.04 seconds is compared to the previous plots. Then, a 20-run, 0.04 second update period run is compared to the three previous plots. Finally, as discussed in Section 4.3.2, $\underline{Q}$ is decreased to 59720 along with an update period decrease to 0.04 seconds to demonstrate the update period effect on the tuning. The results are condensed in Table V-4. Results of Table V-4 indicates first that the SAS program provides results very close (within two to five percent for each state) to the equivalent SSDC run. This can be observed by comparing data columns one and two. Data column number three, when compared to two, indicates that increasing the runs from five to twenty does not significantly change the information that is used in

Table V-4

Equivalency of Simulations
(Mean Errors)

Number	1	2	3	4	5	6
Type	SAS	SSDC	SSDC	SSDC	SSDC	SSDC
Q value	149300	149300	149300	149300	149300	59720
tau(sec)	0.143	0.143	0.143	0.143	0.143	0.143
ud(sec)	0.1	0.1	0.1	0.04	0.04	0.04
Runs	5	5	20	5	20	5
State						
x_1 min	-82	-82.6	-57.2	-26.8	-29.1	-53.8
max	610	647.0	573.5	495.9	475.4	554.5
x_2 min	-112	-110.5	-125.5	-223.4	-216.0	-122.2
max	96	103.3	93.0	60.3	62.3	73.9
x_3 min	-18	-20.6	-9.5	-23.5	-14.4	-12.5
max	94	94.2	88.5	94.0	85.5	89.9
x_4 min	-1400	-1485.0	-1436.0	-1329.5	-1329.5	-1474.4
max	510	514.4	550.0	649.3	645.6	578.4
x_5 min	-100	-96.3	-79.0	-55.4	-57.8	-62.2
max	360	371.0	367.1	611.4	618.4	380.5
x_6 min	-27	-27.7	-15.4	-30.7	-22.0	-16.6
max	65	65.4	62.5	61.1	62.3	62.1
x_7 min	-940	-966.6	-954.3	-1036.9	-1015.5	-1050.7
max	2140	2220.4	2171.1	2047.5	2194.9	2146.8
x_8 min	520	-516.8	-514.2	-674.6	-679.5	-539.1
max	110	100.3	96.8	112.5	136.9	65.5
x_9 min	-28	-28.8	-24.8	-33.9	-26.3	-25.9
max	8	5.8	8.0	22.6	19.3	7.5

ud=update
min=minimum
max=maximum

SAS=Stand-Alone Simulation
SSDC=SOFE/SOFEPL/DISSPLA/Calcomp

performing the tuning process. This justifies running the majority of the simulations at five runs. In other words, the information trend used to vary tuning parameters does not contain any unexpected changes (as also can be observed by studying Figure Sets G.4.1 and G.4.2). It is difficult to quantify the percentage change overall, because different calculation methods lead to different results, and calculations must be done for all nine states. Therefore, to avoid an excessive development on percentage change that is not directly used in this study, it is only required to note than changing from 20 to 5 runs does not cause any unexpected changes in the plotted curve outputs. Data columns four and five again justify the five-run analysis. Additionally, when data columns four and five are compared to columns two and three respectively, they indicate that a significant degree of change occurs in velocity state mean errors (approaching a factor of two for x_2 and x_5) when the update period is changed from 0.1 to 0.04 seconds. To demonstate the effect of the update period change further, data column six is included. For this demonstration, it is decided to decrease the update period back to 0.04 seconds along with a 2.5 times decrease in $\underline{Q}$ and compare the results to the other data columns. Overall, data column six is closer in performance to data columns one through three than with columns four and five. Thus, an "equivalence" is demonstrated, but it is realized the goal to reduce computer costs (Section 4.3.2) or on-line processing loading by increasing the update period from 0.04 t0 0.1 seconds, actually results in the introducion of another tuning parameter. However, this section demonstrates the tradeoffs that must be considered later in the selection of an operational update period (assuming a form of the extended Kalman filter will be implemented on the F-4E/G).

5.3.1.5 Plot Set Five - Filter Comparison

Plot set five compares the tuned preliminary design Kalman filter with six measurements (range, range rate, azimuth angle, elevation angle, azimuth angle rate, and elevation angle rate) without adaptive tuning to the existing Wiener-Hopf filter with six measurements and to a preliminary design Kalman filter with four measurements (range, range rate, azimuth angle, and elevation angle). All filters are tested for the same trajectory, and provide approximately a one-to-one comparison. The SSDC plots included are run at an update period of 0.1 seconds while the Wiener-Hopf filter is run at 0.04 seconds (as shown in the last section an "equivalent" filter can be found at different update periods by adjusting $\underline{Q}$, thus this is not a limiting factor). The results are condensed in Table V-5. Table V-5 indicates the tuned preliminary design Kalman filter significantly outperforms the Wiener-Hopf filter in velocity (approaching a factor of two) and acceleration estimates (Wiener-Hopf position estimates were not provided by OO-ALC). Additionally, the data indicates that a four-measurement update filter can provide comparable filter performance to

Table V-5

Filter Performance
(Mean Errors)

State	Q=149300	Wiener-Hopf	Q=149300(4 meas)
x_1-minimum	-82.6	n/a	-146.9
-maximum	647.0	n/a	590.5
-average	94.4	n/a	56.0
-std dev	190.1	n/a	182.3
x_2-minimum	-110.5	0	-163.4
-maximum	103.3	167	97.3
-average	-22.4	n/a	-24.1
-std dev	51.9	n/a	64.1
x_3-minimum	-20.6	-75	-20.0
-maximum	94.2	100	98.1
-average	44.8	n/a	35.8
-std dev	34.4	n/a	35.3
x_4-minimum	-1485.0	n/a	-1856.2
-maximum	514.4	n/a	552.9
-average	-45.3	n/a	-44.2
-std dev	346.1	n/a	389.5
x_5-minimum	-96.1	-1125	-123.2
-maximum	371.0	750	273.9
-average	71.5	n/a	56.1
-std dev	110.8	n/a	100.9
x_6-minimum	-27.2	-280	24.4
-maximum	65.4	280	68.4
-average	14.1	n/a	11.4
-std dev	20.3	n/a	19.0
x_7-minimum	-966.6	n/a	-1161.0
-maximum	2226.4	n/a	2286.6
-average	235.2	n/a	109.6
-std dev	751.6	n/a	713.5
x_8-minimum	-516.8	-500	-576.9
-maximum	100.3	1450	74.8
-average	-163.1	n/a	-149.4
-std dev	206.5	n/a	212.3
x_9-minimum	-28.8	-280	-25.8
-maximum	5.8	280	2.5
-average	-2.2	n/a	-2.2
-std dev	7.2	n/a	5.7

n/a = not available

that of the six-measurement update filter. This should be considered for actual implementation, as the azimuth and elevation angle rate measurement computations of Equation (3-5) or Equations (3-30) through (3-39) pose a heavy computational burden during filter update. If the processing time or the available memory becomes a serious limitation during implementation, Equations (3-30) through (3-39) can be eliminated and result in performance similar data column three. Finally, note that for all states the average of the mean error is not zero and that the standard deviation is relatively large. This is attributed to the beam trajectory and the Kalman filter performance. During a fighter maneuver, the filter errors are not zero-mean (because of degraded filter performance) as they approach when the fighter is not maneuvering. This is easily observed by studying the plot sets in Appendix G.

5.3.2 Plot Set Six - Tail Chase Simulation

Results from a brief tail chase analysis are condensed in Table V-6. The purpose of the tail chase simulation is to ascertain how the tuned values for the beam attack trajectory perform in the tail chase scenario. Since data on the tail chase performance of the Wiener-Hopf filter was not available from OO-ALC, the tail chase is only studied to gain insight into filter performance. Table V-6 and plot set six indicate the tuned values for the beam attack trajectory result in signicantly biased estimates (for states x_3 through x_9) in

Table V-6

Tail Chase Performace
(Mean Errors)

	Q values (Tau=0.143, update period 0.1 seconds)		
	149,300	373,250	149,300
State	R	R	Rnom
x_1 -minimum	-17.8	-18.0	-22.2
-maximum	43.3	41.5	33.2
-average	-2.1	-2.7	-0.7
-std dev	7.9	8.1	5.9
x_2 -minimum	-24.4	-19.9	-38.8
-maximum	10.4	18.7	10.1
-average	-5.4	-1.9	-4.7
-std dev	7.4	7.9	7.6
x_3 -minimum	-81.1	-107.6	-73.3
-maximum	26.4	58.1	25.5
-average	-38.4	-34.4	-31.1
-std dev	24.2	28.3	22.9
x_4 -minimum	-26.3	-27.3	-13.2
-maximum	312.2	288.7	110.0
-average	144.4	136.4	31.4
-std dev	86.1	85.4	21.5
x_5 -minimum	-126.4	-130.4	-233.4
-maximum	136.4	171.1	69.0
-average	-10.2	-10.5	62.8
-std dev	90.3	81.8	107.7
x_6 -minimum	-2.7	-6.7	-6.3
-maximum	101.4	109.1	106.0
-average	51.1	53.6	62.3
-std dev	26.7	26.3	25.1
x_7 -minimum	-77.1	-73.2	-18.0
-maximum	365.2	38.9	80.9
-average	155.5	169.0	23.5
-std dev	138.4	148.7	25.7
x_8 -minimum	-340.1	-284.1	-416.4
-maximum	34.8	43.4	21.0
-average	-155.0	-127.0	-181.5
-std dev	134.2	109.6	183.0
x_9 -minimum	-66.6	-67.4	-66.7
-maximum	7.0	16.1	15.2
-average	-25.8	-18.6	-11.6
-std dev	22.0	20.2	20.1

the tail chase scenario. The mean error values for states x_3 through x_9 do not return to essentially zero-mean characteristics as in the beam attack because, once the fighter starts its maneuver, it continues maneuvering for the remainder of the simulation. States x_1 and x_2 exhibit good performance since small errors in azimuth and elevation angles have less of an impact on these states due to problem geometry. Better performance for states x_3 through x_9 may be obtained by additional tuning, but at the cost of degrading filter performance during a beam attack. This implies that for implementation, it may be beneficial to change the tuning parameters for the particular trajectory being flown. For example, the range rate could be used as a test to determine which set of tuning parameters to use. If the range rate is high, (head-on attack), moderate (beam attack), or low (tail chase attack) the corresponding best tuning parameters could be used for the filter. Alternately, three filters with different tuning parameters could be implemented simultaneously, in a multiple model adaptive estimator configuration (16:129-136), one for each type of trajectory attack. This requires more computer memory and operating time over a single filter but reduces transient type behavior that results by changing tuning parameters. This option should not be discarded as overly restrictive until further research is completed. It has already been shown that a filter with four meaurements may suffice and that the update period can be increased to 0.1 seconds without serious filter degradation. Thus, it may be possible to implement more than one on-line filter.

5.3.3 Plot Set Seven - Off-Line Adaptive Estimation

At this point, even with increase of performance of the nine-state Kalman filter over the Wiener-Hopf filter, the Kalman filter performance should be improved. The remainder of this chapter addresses ways of possibly improving filter performance while simultaneously searching for the source(s) that cause the filter's degradation. Adaptive estimation, as discussed in Section 3.1, is one method of possibly enhancing filter performance.

The purpose of off-line adaptive estimation in this research is to demonstrate that further research in the area of on-line adaptive estimation is warranted and desirable by establishing a baseline of performance that an on-line adaptive estimation filter can approach. The off-line adaptive estimation plot set, plot set seven, is simulated by lowering $\underline{Q}$ from 373,3250 to 37,325 during a fighter maneuver and conversely raising $\underline{Q}$ to 373,250 when the fighter is not maneuvering. The process of lowering $\underline{Q}$ during a fighter maneuver contradicts the expected tuning process. Normally, during a fighter maneuver, measurement uncertainties increase, and $\underline{Q}$ is increased to compensate for the additional uncertainty (16:121-129). The lowering of $\underline{Q}$ here is attributed to truth model performance (and possibly fighter performance if the truth model accurately models the fighter and measurements during a maneuver). From the truth model output, measurement errors during a fighter maneuver are observed to approach and often exceed the three-sigma values of the $\underline{R}$ matrix. Thus by lowering $\underline{Q}$, more weight is placed on the dynamics model until the measurements become reliable. The SAS program is used for the off-line adaptive estimation because of difficulties that occur in integration routines within SOFE when $\underline{Q}$ was impulsively reset to a different value. The off-line adaptive estimation run is then compared to equivalent nonadaptive SOFE runs with the high and low $\underline{Q}$ values. The results of the simulation are condensed in Table V-7. Table V-7 indicates that adaptive estimation helps in reducing position, velocity, and acceleration errors. Velocity states x_2, x_5, and x_8 for the off-line adaptive estimation have better settling times than those of a constant $\underline{Q}$ of 37,325 and smaller errors then those for a constant $\underline{Q}$ of 373,250. Additionally, settling times of position states x_1, x_4, and x_7 are slightly worse than either of the nonadaptive runs. Finally the acceleration states (x_3, x_6, and x_9) mean error adaptive values fall in between the simulation runs when $\underline{Q}$ equals 37,250 and 373,250. Thus, for the beam attack trajectory, off-line adaptive estimation improves filter performance. Further study in on-line adaptive estimation is warranted and desirable.

Table V-7

Adaptive Estimation Comparison
(Mean Errors / Elapsed Time to Settling)

	(Tau=0.143, update period 0.1 seconds)		
	Q=37,325	Q=variable	Q=373,250
State	SOFE	SAS	SOFE
x_1-minimum	-94.3/8.3[1]	-94/9.2	-57.1/7.4
-maximum	722.5	580	588.4
x_2-minimum	-79.4/12+	-136/11.0	-203.3/8.8
-maximum	111.2	56	99.7
x_3-minimum	-6.8	-38/9.3	-38.3/9.3
-maximum	92.4	95	106.4
x_4-minimum	-1573.9/8.0[1]	-1230/8.5	-1379.6/6.8
-maximum	405.0	600	604.3
x_5-minimum	-87.3/-	-50/11.0	-101.4/9.0
-maximum	210.3	420	620.8
x_6-minimum	-8.4/9.3	-32/9.3	-45.0/9.3
-maximum	65.4	65	83.1
x_7-minimum	-970.2/-[1]	-960/12+	-962.0/12+
-maximum	2308.9	2150	2152.0
x_8-minimum	-361.9/-	-420/12+	-640.3/12+
-maximum	81.7	81	150.7
x_9-minimum	-27.7/6.7	-24/6.7	-31.3/6.7
-maximum	1.7	12	22.3

- = not settled at 12 seconds
+ = almost settled by 12 seconds
1 = biased, mean error value approaches or exceeds the filter-computed covariance value

5.3.4 Isolation of Factors That May Cause Filter Degradation

Even with the increased filter performance obtained through adaptive estimation, overall filter performance enhancement is still desired. It is now beneficial to isolate the cause of filter degradation. Reduced filter performance could be caused by poor measurements, incomplete filter modeling, or incomplete truth state modeling. Each of these are studied in the following subsections.

5.3.4.1 Plot Set Eight - Measurement Lag Removed

The quality of the measurements received from the truth model can cause reduced filter performance. Theoretically, if the dynamic lag of the radar is removed from the simulation, the filter performance should improve. To test this, the trajectory generation program is rerun without radar antenna lag, resulting initially in perfect measurements. Then, either nominal noise, $\underline{R}_{nom}$ of Equation (2-3) or the $\underline{R}$ of Equation (2-33) (the $\underline{R}$ used in the tuned filter) is added to the measurements in SOFE. Data for each of these runs is condensed in Table V-8. Table V-8 indicates the filter does not significantly improve when the dynamic lag of the radar is removed and $\underline{R}$ of Equation (2-34) is used. Using $\underline{R}_{nom}$ of Equation (2-3) is not a fair comparison because $\underline{Q}$ should be changed for tuning and then compared. Thus, misrepresentation of dynamic lags in the measurements can be dismissed as the major cause of filter degradation. This implies the modeling approximations made or the incomplete acceleration state model may now be considered as the cause of reduced filter performance.

Table V-8

Measurement Lag Removed
(Mean Errors)

Q=149300 (Tau=0.143, update period 0.1 seconds)

	no lag	no lag	lag
State	Tuned R	Nominal R	Tuned R
x_1-minimum	-181.5	-227.7	-82.6
-maximum	592.3	284.0	647.0
-average	32.2	20.1	94.4
-std dev	161.2	72.9	190.1
x_2-minimum	-158.3	-409.0	-110.5
-maximum	72.5	98.3	103.3
-average	-25.5	-28.1	-22.4
-std dev	60.4	121.2	51.9
x_3-minimum	-20.1	-21.7	-20.6
-maximum	98.1	101.5	94.2
-average	35.9	37.5	44.8
-std dev	35.4	37.2	34.4
x_4-minimum	-1124.0	-707.6	-1485.0
-maximum	626.1	714.5	514.4
-average	28.5	23.5	-45.3
-std dev	259.5	194.6	346.1
x_5-minimum	-51.1	-276.0	-96.1
-maximum	316.1	1009.7	371.0
-average	66.7	29.7	71.5
-std dev	100.6	218.9	110.8
x_6-minimum	-24.4	-39.0	-27.2
-maximum	68.4	64.2	65.4
-average	11.4	6.1	14.1
-std dev	19.0	20.4	20.3
x_7-minimum	-726.8	-561.0	-966.6
-maximum	1797.2	685.1	2226.4
-average	82.0	7.2	235.2
-std dev	588.1	229.7	751.6
x_8-minimum	-603.9	-1594.2	-516.8
-maximum	158.9	482.6	100.3
-average	-156.4	-178.6	-163.1
-std dev	234.6	471.7	206.5
x_9-minimum	-26.0	-47.4	-28.8
-maximum	4.0	39.9	5.8
-average	2.3	0.0	-2.2
-std dev	5.9	14.9	7.2

5.3.4.2 Plot Set Nine - Acceleration and Truth Model Testing

To support the idea that the first order Gauss-Markov acceleration model may be causing reduced filter performance, the following hypothesis is made. The commonly used value of tau equal to 0.5 seconds in the first order Gauss-Markov zero-mean acceleration model for fighter type targets (17) does not provide adequate performance. A tau of 0.143 seconds provides superior performance. In other words, to compensate for incomplete modeling, tau is changed. The value of tau affects filter performance most when the fighter maneuvers. Thus the value of tau which is part of the target acceleration model is changed to compensate for fighter acceleration, a compensation for incomplete modeling.

To test the above hypothesis, two new beam trajectories are generated. First the original beam trajectory is modified so there are no fighter maneuvers. Then, the original trajectory is modified again so the target does not accelerate. In this manner, it is possible to isolate whether incomplete target acceleration modeling or the truth model is causing reduced filter performance. The results of SSDC runs using the above described trajectories are condensed in Table V-9. The results of Table V-9 and the associated plots show that the filter performs well when there are no fighter maneuvers (based on the minimal velocity and settling performance criteria of Sections 4.5 and 5.3.1.2). On the other hand, the filter reverts to degraded performance when the fighter rolls and the target does not

Table V-9

Model Testing
(Mean Error)

Q=149300 (Tau=0.143, update period 0.1 seconds)

State	no fighter man	no target man	both man
x_1-minimum	-24.4	-162.4	-82.6
-maximum	130.9	538.0	647.0
-average	38.9	43.9	94.4
-std dev	41.2	170.2	190.1
x_2-minimum	-33.1	-196.0	-110.5
-maximum	104.7	18.8	103.3
-average	33.9	-59.7	-22.4
-std dev	37.1	71.7	51.9
x_3-minimum	-19.7	-26.4	-20.6
-maximum	94.2	34.4	94.2
-average	41.8	1.4	44.8
-std dev	33.2	37.2	34.4
x_4-minimum	-131.7	-1432.9	-1485.0
-maximum	46.4	629.4	514.4
-average	-37.6	2.8	-45.3
-std dev	40.8	353.8	346.1
x_5-minimum	-89.2	-35.4	-96.1
-maximum	28.9	424.9	371.0
-average	-23.5	92.7	71.5
-std dev	31.1	126.3	110.8
x_6-minimum	-27.3	-21.5	-27.2
-maximum	72.9	30.4	65.4
-average	16.7	0.7	14.1
-std dev	22.6	9.2	20.3
x_7-minimum	-80.5	-995.6	-966.6
-maximum	150.6	2181.2	2226.4
-average	33.0	216.3	235.2
-std dev	53.0	724.3	751.6
x_8-minimum	-33.3	-527.1	-516.8
-maximum	54.0	40.0	100.3
-average	8.2	-175.4	-163.1
-std dev	19.6	194.2	206.5
x_9-minimum	-2.0	-15.4	-28.8
-maximum	2.6	8.8	5.8
-average	0.2	0.0	-2.2
-std dev	0.9	3.9	7.2

accelerate. This implies either the rotations before the measurement update (see Section 4.3.3 and Appendix D) or the truth model is causing filter degradation. The rotations used are dismissed as suspect for several reasons. First, they have been checked for proper implementation. Second, after the fighter stops maneuvering, the filter recovers, which indicates some degree of correctness. Third, a concurrent research effort studying the F-4E/G long range intercept problem (4) using a completely inertial model (thus avoiding the rotations in question) also experiences similar filter degradation when using an equivalent truth model. Finally, the Wiener-Hopf filter is experiencing a similar problem, in both simulation and actual implementation. The one common simulation element is the truth model. Thus, as a next logical step, it is recommended that the truth model be revalidated, to ensure a problem has not been designed into the truth model that is causing the simulated Kalman filter degradation. Alternatively, a test flight could be flown using the nine-state Kalman filter to determine if the problem exists in the real world for the Kalman filter as it does for the Wiener-Hopf filter. If it is determined the problem is real for the Kalman filter, an ad hoc procedure based on the measurement residuals that may enhance the nine-state EKF performance is suggested for future research in the next chapter.

5.4 Simulation Implications

Overall, this chapter demonstrates that the simulation results of the preliminary design nine-state extended Kalman filter exceeds the performance of the current Wiener-Hopf filter. As such, the preliminary design should be continued or expanded into a final design. The next chapter summarizes the preliminary design and recommends steps that should be considered in developing the final design.

VI. CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions

6.1.1 Problem Review

Currently, the F-4E/G uses a Wiener-Hopf filter for estimating target position, velocity, and acceleration during air combat maneuvering. As implemented, the errors between the actual target variables and the estimate of these variables are too large. *The purpose of this study is to* evaluate the feasibility of replacing the Wiener-Hopf filter with a Kalman filter in order to obtain better estimates. The evaluation is made by first designing an appropriate Kalman filter and then testing the design through computer simulation. The computer simulation results indicate that the Kalman filter significantly outperforms the Wiener-Hopf filter. Thus, the Kalman filter is a feasible choice for replacing the Wiener-Hopf filter.

6.1.2 Design Review

The extended Kalman filter developed is a preliminary design for making the evaluation described above. The Kalman filter contains nine states (three relative target position, three total target velocity, and three total target acceleration states). Filter propagation is based on linear time-invariant dynamics primarily because of the limited capabilities of the on-board aircraft computer. The linear dynamics permits propagation by a state transition matrix, resulting in a computationally efficient implementation. Measurement updates use six measurements (range, range rate, azimuth angle, elevation angle, azimuth rate, and elevation rate) available on the F-4. An extended Kalman filter is used since nonlinear measurement equations result when the measurments are expressed in terms of the states. Both continuous time sampled-data and discrete-time sampled-data designs are included.

6.1.3 Results Review

The continuous time sampled-data design is used in validating filter performance through a Monte Carlo analysis using a computer aided design package called Simulation for Optimal Filter Evaluation (SOFE). The equivalent discrete-time design is also used in validating filter performance, and, since it is a discrete-time design, it can be easily implemented on the F-4E/G computer. The results of the simulation indicate the following:

1. The current update period of 0.04 seconds can be increased to 0.1 seconds with filter retuning and still retain "near equivalent" filter performance (see Section 5.3.1.4 and Table V-5).

2. The discrete-time design and simulation provide results very similar to the results from SOFE for the same problem parameters (within two to five percent for all states, see Table V-5).

3. The Kalman filter significantly outperforms the Wiener-Hopf filter for the beam attack trajectory

tested. Position state information is not available for the Wiener-Hopf filter, but for velocity and acceleration, overall performance is increased by more than a factor of two as shown in Table V-4. Other trajectories are not directly compared since Wiener-Hopf filter data was not available.

4. A Kalman filter using a four-measurement filter update (range, range rate, azimuth angle, and elevation angle) also significantly outperforms the Wiener-Hopf filter. Again, for velocity and acceleration states, overall peformance is improved by more than a factor of two. Compared to a 6-measurement filter, overall position error increases 11 percent. As explained in Section 5.3.1.5, a four-measurement filter reduces the computational loading and may allow more than one on-line filter.

5. The tuned filter for a beam attack provided biased estimates for a tail chase scenario (see Table V-6 and pages G-172 through G-180). Thus, as explained in Section 5.3.2, more than one on-line filter may be desirable.

6. Results of off-line adaptive estimation simulations show on-line adaptive estimation can provide substantially enhanced performance over a nonadaptive filter (see Table V-7).

7. To reduce state estimate errors further, the final tuned filter's performance should be improved. Possible sources of filter degradation are poor measurements, modeling approximations, or errors in the truth model. Measurements and modeling approximations are tested by an appropriate choice of simulations and are found not to be a major source of degradation. As a result, doubt is cast on the truth model performance which implies, as described in Section 5.3.4.2, that the trajectory generation (truth model) should be revalidated.

6.2 Recommendations

The preliminary filter designed in this thesis should be pursued as viable replacement for the Wiener-Hopf filter. Areas where additional study should be concentrated to further develop the Kalman filter can be divided into tuning, remodeling and testing categories.

6.2.1 Tuning

To enhance the Kalman filter performance further, additional tuning should be considered. Specific areas where additional tuning may provide enhancement are:

1. Tuning on a particular aircraft body axis (i_o, j_o, k_o). Because of the problem and aircraft geometry, for the trajectories tested, the target parameters in i_o change much slower than either j_o or k_o (an aircraft's roll response is much more rapid than heading response

in a turn). This implies further testing using different values of Q and tau for different axes may be beneficial.

2. Tuning for several different trajectories (beam attack, tail chase, and head-on attacks).

3. If possible, development of an overall nonadaptive tuned filter suitable for all trajectories.

4. Development of an on-line adaptive filter to tune adaptively to trajectory changes.

6.2.2 Testing

Beyond the tuning described above, further testing is required. Areas where testing may be beneficial are:

1. Use of a different trajectory generation simulation (truth model) to determine if filter performance inproves.

2. Implementation of the existing discrete-time design with a modified update algorithm (using a U-D update algorithm as discussed in Section 4.4.2) on an OO-ALC test aircraft and a flight test performed to verify the filter's performance in the real world.

3. Addition of noise to the inertial measurement unit (IMU) information, to determine the affect of IMU noise on the Kalman filter performance. Chapter V simulation

is based on a perfect IMU. Adding noise to the IMU information will affect both filter propagation by adding noise to fighter velocity terms in Equation (2-10) as well as filter update equations by adding noise to rotation terms used in performing the transformations just prior to the measurement update.

4. Additional simulation of the equivalent discrete-time design using a 16-bit, fixed point, word-length configuration to determine the wordlength effect on the Kalman filter performance. The discrete-time filter update algorithm should be changed to a U-D algorithm for numerical stability.

6.2.3 Remodeling

Remodeling may be required if the above retuning and additional testing steps result in performance less than desired. Adding two states to the propagation model for estimating the antenna azimuth and elevation errors may be necessary if these error signals can not be obtained from the aircraft radar. But, as discussed in paragraph 6.1.3, item 7, the present simulated "measurements" do not appear to cause serious filter degradation.

Finally, ad hoc procedures may be evaluated for achieving enhanced filter performance. One proposed procedure is based on noting that, when the fighter maneuvers, the state errors become large and the measurement residuals for angles and angle rates cross over their filter-computed one-sigma values. If a fighter maneuver is detected (from the IMU accelerometers or fighter control stick position), the position, velocity, and acceleration estimates can be modified by calculated increments and $\hat{\underline{x}}(t_i)$ recomputed as explained below. Using residual information when it is outside an established one-sigma bound, the corresponding amount over the one-sigma can be added back into the problem geometry as unmodeled displacements, velocities and accelerations. A position correction factor for j_o and k_o can be calculated from the geometry using the i_o position estimate and the residual angle over the one-sigma bound, a velocity correction term calculated by dividing the position correction by the update period, and an acceleration correction by dividing the velocity correction term by the update period. This is not unreasonable, since, as shown in the simulation, position errors in i_o are much smaller than j_o and k_o. Additionally, in the next propagation cycle, all three axes will be affected in performing the calculation of $\underline{H}[t_i;\hat{\underline{x}}(t_i^-)]$. The recomputed $\hat{\underline{x}}(t_i)$ would then be equal to the old $\hat{\underline{x}}(t_i)$ plus the calculated correction factor. Increased computational load would only occur when the residuals are outside the one-sigma bound. Thus, with minimal calculations and additional computer loading, unmodeled affects may be compensated for using this ad hoc proposal. Other ad hoc proposals that may be beneficial are outlined in a paper by Maybeck, Jensen, and Harnly (24). In particular, this reference found that using

$[\hat{x}(t_i^+)-\hat{x}(t_i^-)]$ to recalculate state estimates could serve as a model correction to account for both filter bias and maneuver detection of the target. Both the above approaches should be studied to determine which one provides the most performance enhancement. Then, based on the degree of performance enhancement, it should be decided if the additional states are required (assuming the angle errors can not be obtained from the radar control unit).

Bibliography

1. Robinson, Clarence A. Jr. "Defense Dept. Backs F-4 Upgrade Program", Aviation Week & Space Technology, 120 (2): 16-18 (January 9, 1984).

2. Hougaard, Hugh, Sponsoring Engineer. "Proposed AFIT Thesis Topic", Research Request by OO-ALC/MMECB, Hill AFB, UT: 1-2 (December, 1984).

3. Steering Dot Stability, Unpublished Notes, OO-ALC/MMECB Hill AFB, UT (date unknown).

4. Halbert, Robert C., Kalman Filter Design for the Long Range Intercept Function of the F-4E/G Fire Control System, M.S. thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH (December 1985).

5. Maybeck, Peter S., Stochastic Models, Estimation, and Control, Vol. 1. New York: Academic Press, Inc., 1979

6. Weston, Andrew C., Dual-Seeker Measurement Processing for Tactical Missile Guidance, M.S. thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1982, (ADA124725).

7. Landau, Mark I. "Radar Tracking of Airborne Targets", Paper presented at National Aerospace and Electronics Conference: Dayton, OH, 19 May 1976.

8. F-4E/G Operational Flight Program (OFP) Description, P004 Update, OO-ALC/MMECB, Hill AFB, UT, 27 January 1984.

9. Barton & Ward, Handbook of Radar Measurement, New Jersey: Printice-Hall, Inc., 1969.

10. User's Manual for OFP for F-4E ACM Computer Class V Modification 2745, P004 Update, OO-ALC/MMECB, Hill AFB, UT, 27 January 1984.

11 A Digital LRU-1 Input Noise Model, Unpublished Notes, OO-ALC/MMECB, Hill AFB, UT (date unknown).

12. Trajectory Generation Computer Simulation, Truth Model, Computer Program, OO-ALC/MMECM (date unknown)

13. Worsley, William H, Comparison of Three Extended Kalman Filters for Air-to-Air Tracking, M.S. thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1980, (ADA094767).

14. Bryan, Ralph S., Cooperative Estimation of Targets by Multiple Aircraft, M.S. thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, June 1980, (ADA085799).

15. Beal, Bob and Chuck Strickler, OO-ALC/MMECB. Telephone Conference. Hill AFB, UT, 1 October, 1985.

16. Maybeck, Peter S., Stochastic Models, Estimation, and Control, Vol. 2. New York: Academic Press, Inc., 1979

17. Maybeck, Peter S., Lecture notes in EE 766, Stochastic Estimation and Control II, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, May 1985.

18. Musick, Stanton H., "SOFE: A Generalized Digital Simulation for Optimal Filter Evaluation, User's Manual", Report Number AFAL-TR-80-1108, October 1980, (ADA093887).

19. Musick, Stanton H., Richard E. Feldman, and Jerry G. Jensen, "SOFEPL: A Plotting Postprocessor for 'SOFE', User's Manual", Report Number-TR-80-1109, November 1981, (ADA111923).

20. Computer Graphics Package, DISSPLA, Integrated Software Systems Corporation, San Diego, CA, (1981).

21. Computer Plotting Package, Calcomp, California Computer Products, Inc., Anaheim, CA, (1982).

22. Beal, Bob, Project Engineer. Conference Meeting at Wright-Patterson AFB, OO-ALC/MMECB, Hill AFB, UT, (September 1985).

23. Beal, Bob, Project Engineer. "Wiener-Hopf Target Velocity and Acceleration Plots", Official Correspondence, OO-ALC/MMECB, Hill AFB, UT, (17 September 1985).

24. Maybeck, Peter S., R.L. Jensen, and D.A. Harnly, "An Adaptive Extended Kalman Filter for Target Image Tracking", IEEE Trans. on Aerospace and Electron. Sys., Vol. AES-17, No. 2: 173-180 (March 1981).

APPENDIX A

LITERATURE REVIEW FOR ACM

I. ARTICLES

1. Singer, Robert A., "Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-6, (4): 473-483 (July 1970).

   Abstract: The majority of tactical weapons systems require that manned maneuverable vehicles, such as aircraft, ships, and submarines, be tracked accurately. An optimal Kalman filter has been derived for this purpose using a target model that is simple to implement and that represents closely the motions of maneuvering targets. Using this filter, parametric tracking accuracy data have been generated as a function of target maneuver characteristics, sensor observation noise, and data rate and that permits rapid a priori estimates of tracking performances to be made when maneuvering targets are to be tracked by sensors providing any combination of range, bearing, and elevation measurements.

2. Fitts, John Murray, "Aided Tracking as Applied to High Accuracy Pointing Systems," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-9 (3): 350-368 (May 1973).

   Abstract: Two basic concepts of rate aided tracking and position aided tracking are applied to a conventional pointing system in order to improve performance. The aided track signals are derived in an inertial space format and are generated from a Kalman filter algorithm. Computational results are included to show the interplay between the conventional pointing system and the aided track filter.

3. Pearson, John B., Edwin B. Stear, "Kalman Filter Applications in Airborne Radar Tracking," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-10 (3): 319-329 (May 1974).

   Abstract: This paper studies the application of Kalman filtering to single-target track systems in airborne radar. An angle channel Kalman filter is configured which incorporates measures of range, range rate, and on-board dynamics. Theoretical performance results are given and a discussion of methods for reducing the complexity of the Kalman gain computation is presented.

A suboptimal antenna controller which operates on the outputs of the angle Kalman filter is also described. In addition, methodological improvements are shown to exist in the design of range and range-rate trackers using the Kalman filter configuration.

4. Farrel, James L., Elmen C. Quesinberry, Charles D. Morgan, and Michael Tom. "Dynamic Scaling for Air-to-Air Tracking," Proc. Nat. Aerospace and Electron. Conf., Dayton, Ohio: 157-162 (May 1975).

   Abstract: This paper presents an air-to-air tracking approach providing accurate estimates of position and velocity vectors, plus target acceleration which enhances fire control and track in blind conditions (e.g., main beam clutter) or severe dynamic conditions (e,g., close range, high-g maneuvers). Even with a host of degradations at realistic levels, accuracies on the order of 30 ft, 2 mrad, and 3 mrad/sec are obtained for range, LOS, and LOS rate, respectively, in steady-state track. Transient performance is characterized by monotonic reduction of lock-on errors and suppression of disturbances. The system integrates readily with all INS and radar data, while fully capitalizing on all processing performed with no loss of efficiency. Complete flexibility of mechanization is easily exploited to provide backup in the event of INS failure.

5. Landau, Mark I., "Radar Tracking of Airborne Targets," Proc. Nat. Aerospace and Electron. Conf., Dayton, Ohio: (May 1976).

   Abstract: Airborne radar target trackers are required to provide accurate estimates of target motion for radar and fire contrcl functions. Precise target tracking accuracies can be achieved through the use of modern estimation techniques. This paper examines a Kalman filtering approach to airborne tracker design. Alternative tracking configurations for a single target track environment are presented, and the characteristics of each configuration are discussed. Tracking performance of one of the configurations is also presented to illustrate typical target tracking accuracies.

6. Tenney, Robert R., Ralph S. Hebbert, and Nil R. Sandell, "A Tracking Filter for Maneuvering Sources," IEEE Transactions on Automatic Control: 246-251 (April 1977).

   Abstract: It is well known that the extended Kalman filtering methodology works well in situations characterized by a high signal-to-noise ratio, good observability and a valid state trajectory for

linearization. This paper considers a problem not characterized by these favorable conditions. A large number of ad hoc modifications are require to prevent divergence, resulting in a rather complex filter. However, performance is quite good as judged by comparison of Monte Carlo simulations with the Cramer-Rao lower bound, and by the filters ability to track maneuvering targets.

7. Gholson, Norman H., Richard L. Moose, "Maneuvering Target Tracking Using Adaptive State Estimation," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-13 (3): 310-317 (May 1977).

Abstract: Two approaches to a nonlinear state estimation are presented. The particular problem addressed is that of tracking a maneuvering target in three-dimensional space using spherical observations (radar data). Both approaches rely on semi-Markov modeling of target maneuvers and result in effective algorithms that prevent the loss of track that often occurs when a target makes a sudden, radical change in its trajectory. Both techniques are compared using real and simulated radar measurements with emphasis on performance and computational burden.

8. Moose, R.L., H.F. Vanlandingham, and D.H. McCabe, "Modeling and Estimation for Tracking Maneuvering Targets," IEEE Transaction on Aerospace and Electronic Systems, Vol. AES-15 (3): 448-456 (May 1979).

Abstract: A new approach to the three-dimensional airborne maneuvering target tracking problem is presented. The method, which combines the correlated acceleration target model of Singer (3) with the adaptive semi-Markov maneuver model of Gholson and Moose (8), leads to a practical real-time tracking algorithm that can be easily implemented on a modern fire-control computer. Preliminary testing with actual radar measurements indicates both improved tracking accuracy and increased filter stability in response to rapid target accelerations in elevation, bearing and range.

9. Maybeck, Peter S.,William H. Worsley and Patrick M. Flynn. "Investigation of Constant Turn-Rate Dynamics Models in Filters for Airborne Vehicle Tracking," Proc. Nat. Aerospace and Electron. Conf., Dayton, Ohio: 896-903 (May 1982).

Abstract: A constant turn-rate model for acceleration has been proposed as more representative of airborne vehicle motion characteristics than models currently employed, such as first order Gauss-Markov or Brownian

motion. Target trackers in the form of extended Kalman filters based on these alternative dynamics models are developed and analyzed for both air-to-air gunnery (estimation in three dimensions) and FLIR image focal plane target intensity tracking (in two dimensions). In both applications, the filter based on the constant turn-rate model displays smalller estimation biases, indicative of a better internal model within the filter structure, but a moderate performance enhancement is offset by a significant increase in computational loading.

10. Asseo, Sabi J. and Richard J. Ardila. "Sensor-Independent Target State Estimator Design and Evaluation," Proc. Nat. Aerospace and Electron. Conf., Dayton, Ohio: 916-924 (May 1982).

Abstract: ...develop an estimator whose structure is invariant with sensor configuration and adaptable to various target maneuvers. .... Two alternate estimator designs are developed herein by using local-level inertial (Cartesian) coordinates and line-of-sight (spherical) coordinates. In each estimator, target accelerations are represented as a second-order-dependent coefficients which adapt automatically to target maneuvers. The performance of these estimates is evaluated in simulated air combat. Both estimators produce unbiased estimates of target acceleration, more accurate velocity and range estimates, and a smaller miss distance than a typical state-of-the-art estimator.

11. Scharf, Louis L., Sigurdur Sigurdsson, "Fixed Point Implementation of Fast Kalman Algorithms," Report to Office of Naval Research under contract N00014-82-K-0300, Arlington, VA (November 1983).

Abstract: In this páper we study scaling rules and round-off noise variances in a fixed point implementation of the Kalman predictor for an ARMA time series observed noise-free. The Kalman predictor is realized in a fast form that uses the so-called fast Kalman gain algorithm. The algorithm for the gain is fixed point.

Scaling rules and expressions for rounding error variances are derived. The numerical results show that the fixed point realization performs very closely to the floating point realization for relatively low-order ARMA time series that are not too narrowband.

The predictor has been implemented in 16-bit fixed point arithmetic on an INTEL 8086 microprocessor, and in 16-bit floating point arithmetic on an INTEL 8080. Fixed point code was written in ASSEMBLY language and floating point code was written in FORTRAN. Experimental results were obtained by running the fixed and floating point filters on identical data sets. All experiments were carried out on an INTEL MDS 230 Development System.

12. Chang, Chaw-Bing and John Tabaczynski. "Application of State Estimation to Target Tracking," IEEE Transactions on Automatic Control, Vol. AC-29 (6): 98-109 (February 1984).

Abstract: In this paper, we present a survey of problems and solutions in the area of target tracking. The discussion includes design tradeoffs, performance evaluation, and current issues.

II. THESES

1. Kolibaba, R.L., Precision Radar Pointing and Tracking Using an Adaptive Kalman Filter. M.S. thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio(January 1973).

Abstract: An Adaptive Extended Kalman Filter technique was developed to improve the tracking capabilities of an airborne tracking system. A maneuver determination technique was developed as well as an adaptive technique for the probability model of target acceleration. The computer simulation was run using four different types of data. Computation of the acceleration state was inaccurate due to imprecise modeling of the acceleration states. The adaptive technique reduced the range error of the non-adaptive filter. A maneuver was determined when a bias was detected on the range measurement residual.

Kalman Filter(s): Adaptive and extended.

Radar Model(s): Not determined.

Trajectory Generation: Target vehicle only, x-y plane only (see page 29).

Acceleration Model: Two triangular peaks (see page 20).

DTIC Number: AD 768378

2. Lindberg, E.K., A Radar Error Model and Kalman Filter for Predicting Target States in an Air-to-Air Environment. M.S. thesis, Air Force Institute of

Technology, Wright-Patterson AFB, Ohio(December,1974).

Abstract: Various extended Kalman filters and modified nonlinear filters are examined for target acceleration estimation over a representative trajectory. The filters are of two basic types. The first type uses the radar measurements in the antenna centerline coordinate frame to provide the Kalman filter measurements. The propagation equations for this implementation have position, velocity, and acceleration as the states in a convenient inertial reference frame. In comparison, the second type performs propagation and measurement equations computations entirely in the antenna centerline frame.

Each type of filter is tested for any degradation of performance apparent from excluding radar measurements of the error angles between the antenna centerline and line-of-sight reference frame.

Due to the simplicity and accuracy of the extended Kalman filter of the first type that does not include measurements of the error angles, that filter is then chosen for further evaluation. This testing involves another trajectory and includes the task of estimating relative velocity, position, and acceleration of the target.

Another outcome of the thesis is a radar error model representative of the 1960 generation of air-to-air radars. This analytically useful result includes appropriate stochastic models of the inaccuracies of the radar, heretofore unavailable in such system descriptions.

The results demonstrate that the filter is extremely sensitive to certain maneuvers. A graphic filter performance analysis is included that displays this sensitivity and portrays overall filter performance.

Kalman Filter(s): Extended

Radar Model(s): Azimuth, elevation (decoupled assumption), and range.

Trajectory Generation: Model by McDonnell Aircraft Company. Expo IV, Evaluation of Fixed Base Tracking Simulation. St. Louis, Mo.: 16 March 1973.

Acceleration Model: 1st order Gauss-Markov (see pages 24-32).

DTIC Number: AD-A008671

3. Lutter, R.N., Application of an Extended Kalman Filter to an Advanced Fire Control System. M.S. thesis, Air Force Institute of Technology, Wright-Patterson AFB,

Ohio (December,1976).

Abstract: An extended Kalman Filter is developed to aid the tracking of an air-to-air missile from a maneuvering target aircraft. The filter exploits knowledge of the dominant dynamic effects acting on a missile that is non-thrusting and utilizing a proportional navigation guidance scheme, i.e. accelerations due to aerodynamic forces. It is designed to provide both dynamic tracking estimates in a local inertial frame and estimates of the proportional navigation constant and another pertinent parameter.

A feasibility analysis of the filter is conducted. Its performance is compared to a more conventional filter that utilizes a first order Gauss-Markov random process acceleration model. In addition, an evaluation is made of the filter's capability to recover from large initial errors in state estimates.

The study establishes the feasibility of the modelling approach. The estimates provided by the designed filter are, in general, less sensitive to system measurement noises. The filter performance is trajectory dependent, however, and the requirement for a higher order missile model within the filter system model is established (a zero-order model was used to develop as simple a filter as would provide adequate performance).

The results of the study strongly suggest that the navigation constant can be estimated by the filter. The recovery analysis provides additional insights into the filter's ability to estimate this parameter. It gives a general indication of the effects that varying the initial variance and noise strength (on the navigation constant channel) have on the tuning and recovery characteristics of the navigation constant estimate. A graphic filter analysis is included that portrays the estimation accuracy and recovery characteristics of the filter.

Kalman Filter(s): extended

Radar Model(s): Gimballed platform with rate gyros (see pages 9-50).

Trajectory Generation: TRAJ

Acceleration Model: Based on dynamics.

DTIC Number: AD A035293

4. Ryan, J. E., Sensitivity Study or Strapdown Inertial Sensors in High Performance Applications, M.S. thesis, Air Force Institute of Technology, Wright-Patterson AFB,

Ohio (December 1980)

Abstract: This study uses a computer simulation of a strapdown laser gyro inertial reference system to analyze the errors generated as a result of highly dynamic flight profiles. A stochastic error model using state-of-the-art inertial sensors is developed in detail and implemented in software. SOFE, a generalized simulation program, was used to implement both a Monte Carlo simulation and a covariance analysis. The Monte Carlo method was selected to perform the error analysis.

Two highly dynamic flight trajectories were developed using the flight profile generator, PROFGEN. The PROFGEN program itself was modified to include an aircraft roll time constant and a roll-only maneuver. The errors generated in the inertial reference system as a result of these flight trajectories were investigated. Both an error budget and an analysis of the maneuvers inducing these errors were accomplished.

Gyro error sources induced the most system error and coupled the dynamics of the flight trajectory into the variations of the error. Misalignment was found to be the major cause of both the accelerometer and gyro induced error. Successive maneuvers were found that reinforced system errors and other maneuvers were found that cancelled these errors. Also, some cases were found where the amount of system error varied with a change in heading.

Kalman Filter(s): not determined

Radar Model(s): not determined

Trajectory Generation: PROFGEN (modified for roll only)

Acceleration Model: not determined

DTIC Number: AD A100825

5. Bryan, R.S., Cooperative Estimation of Targets by Multiple Aircraft. M.S. thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio (June 1980).

Abstract: (not copied)

Kalman Filter(s): not determined

Radar Model(s): Simplified (see pages 8-18).

Trajectory Generation: Used, but not included. Thrust and roll assumed to be instantly achieved, producing unrealistic step changes in target acceleration (see page 51).

Acceleration Model: First order Gauss-Markov and constant turn rate (see page 8).

DTIC Number: AD A085799

6. Worsley, W.H., Comparison of Three Extended Kalman Filters for Air-to-Air Tracking. M.S. thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio (December 1980).

Abstract: The performances of the extended Kalman filter implementations for three different target acceleration models that estimate target position, velocity, and acceleration states for air-to-air gunnery were compared. The models included 1) a first order zero-mean Gauss-Markov relative target acceleration model, 2) a first order zero-mean Gauss-Markov total target acceleration model, and 3) a constant turn rate target acceleration model. Measurements available to the extended Kalman filter at update were the range, range rate, and the error angles between the true line of sight and the estimated line of sight. Additional evaluations of the effect of variations in the variances of measurement noises were conducted for the extended Kalman filter using the constant turn rate target acceleration model. All evaluations were accomplished using Monte Carlo simulation techniques.

Kalman Filter(s): extended

Radar Model(s): Tracker dynamics not included in the filter model, assumed tracker moves to new position without error before next measurement (see pages 24,29). Assumed to be inertially space-stabilized.

Trajectory Generation: TRAJ

Acceleration Model: See abstract.

DTIC Number: AD A094767

7. Flynn, P.M., Alternative Dynamics Models and Multiple Model Filtering for a Short Range Tracker. M.S. thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio (December 1981)

Abstract: The performance of three extended Kalman filter implementations that estimate target position, velocity, and acceleration states for a laser weapon system are compared using various target acceleration trajectories. Measurements available to the extended Kalman filters each update are taken directly form the outputs of a forward looking infrared (FLIR) sensor. Two dynamics models considered for incorporation into the filter are 1) a Brownian motion (BM) acceleration and 2) a constant turn rate (CTR) target dynamics model. The CTR filter was compared against the BM filter to see if the more complex dynamics of the CTR filter gave it a significant improvement in tracking performance over the BM filter. These two simple extended Kalman filters were then compared to a multiple model adaptive filter consisting of a bank of three filters based on the Brownian motion acceleration model. All three filter are tested using three different flight trajectory simulations: a 2 g, a 10 g and a 20 g pull up maneuver. All evaluations are accomplished using Monte Carlo simulation techniques.

The constant turn rate extended Kalman filter was found to outperform the other two filters. The main advantage this filter had was the minimization of mean bias error in estimating position. The standard deviation of error was also slightly lower in most instances.

Kalman Filter(s): extended

Radar Model(s): not determined

Trajectory Generation: 2, 10, and 20g pull ups.

Acceleration Model: see abstract

DTIC Number: AD A115503

Appendix B

Trajectory Generation Program (Truth Model)

```
     PROGRAM HILL5
C
C****** APQ-120 RADAR MODEL--RADAR.FOR *******
C      VERSION 1
C
C      DESCRIPTION:  THIS PROGRAM DRIVES THE APQ-120 RADAR MODEL,
C      AIRCRAFT/TARGET GEOMETRY MODEL AND FILTER GIVEN INITIAL
C      CONDITIONS AND TIME STEP CONDITIONS.
C      *THIS PROGRAM REQUIRES SUPPORT ROUTINES DEFINED IN THE FILES
C      RADAR.FTN AND ROTATE.FOR.
*****BOTH RADAR.FTN AND ROTATE.FOR ARE INCLUDED IN THIS VERSION
*****OF PROGRAM HILL5.
C      *INPUTS TO THIS PROGRAM ARE CONTAINED IN A TEST DEFINITION
C      FILE INTEST (FORMERLY TEST1.DAT)
C
      COMMON/MAIN/TINL,TFNL,NNT,NDIFF,NTSOUT,
    1    NDATA,NOUT
      COMMON/STEP/T,DT,DT2
      COMMON/RADAR/Y(2,5),Y1(2,5),Y2(2,5),X1(2,2),OUT(2,2),
    1    A,B,C,D,E,GA,KA,K,COEF1,COEF2,COEF3,COEF4,COEF5,COEF6,
    2    XX(3),YY(3),MODE,IRADAR
*****ADDED TANG***************************************************
      COMMON/AIR/RATE,PHI,PHIDT,PPHI,PITCH,PITCHD,PPITCH,TRATE,
    1    TURN,TURND,PTURN,TR1,TR2,TR3,TR4,TF1,TF2,TF3,TF4,ANGLE,
    2    ASPECT,RANGE,RDOT,RANGEP,AOA,TAS,TAST,ALPHA,BETA,ALPHA1,
    3    ALPBET,BETA1,XFDIST,YFDIST,XTDIST,YTDIST,XTCON,YTCON,
    4    HORT1,HORT2,HORTG,TGEES,DELTA,IAIR,TANG
*****AT,VTIJKO,ATIJKO ADDED****************************************
      COMMON/FILTER/VT(3),VF(3),VR(3),NFIL,IFILT,AZ,EL,WK,WJ,ROLL,
    1    YAW,YNAZ,YNEL,YNWK,YNWJ,YNROLL,YNYAW,FR,SN1,SN2,SN3,CONLP1,
    2    CONLP2,PI,I1,I2,U1,U2,V1,V2,VT1(3),VT2(3),
    3    AT(3),VTIJKO(3),ATIJKO(3)
      COMMON/RK4SAV/SAV(2,5)
      LOGICAL NFIL
      REAL K,KA
C      DIMENSION NAME(30)
C      BYTE NRESP
*****THE FOLLOWING 2 LINES ARE USED IN OUTPUTTING FORM 3, ADDED CODE***
********DIMENSION XSXYZ(3),XSTURN(3),XNED(3),XXYZ(3),XIJKO(3)
     DIMENSION VTIJK(3), VTIJKAZ(3)
      CHARACTER TITLE3*40,MODE3*20,NAME*30
      DATA Y/10*0./, Y1/10*0./
C
C      WRITE(5,900)                        !PROGRAM BANNER
      WRITE(*,900)
C
C****** SELECT OUTPUT FORMAT AND LOAD OFP *******
C
C2     WRITE(5,*)'SELECT OUTPUT FORMAT AS FOLLOWS:'
2      WRITE(*,*)'SELECT OUTPUT FORMAT AS FOLLOWS:'
C      WRITE(5,*)'FORMAT 0: T,ROLL,AZ,EL,WK,WJ,TURND,PHIDT,PHI'
      WRITE(*,*)'FORMAT 0: T,ROLL,AZ,EL,WK,WJ,TURND,PHIDT,PHI'
C      WRITE(5,*)'FORMAT 1: T,FLAG,VFI,J,K,VTI,J,K,TGEES'
```

```
        WRITE(*,*)'FORMAT 1: T,FLAG,VFI,J,K,VTI,J,K,TGEES'
C       WRITE(5,*)'FORMAT 2: FORMAT 0 + FORMAT 1'
        WRITE(*,*)'FORMAT 2: FORMAT 0 + FORMAT 1'
*****ADDED OUTPUT FOR SOFE*******************************************
        WRITE(*,*)'FORMAT 3:OUTPUT FOR SOFE, 6 MEAS, TRUTH STATES, ETC'
*****END OF ADDED CODE***********************************************
C       READ(5,*)NOUT
        READ(*,*)NOUT
C       IF(NOUT.LT.0.OR.NOUT.GT.2)THEN
        IF(NOUT.LT.0.OR.NOUT.GT.3)THEN
           WRITE(5,*)'FORMAT NUMBER DOES NOT EXIST'
           GO TO 2
        ENDIF
C
C****** INPUT TEST DEFINITION FILE *******
C
C5      OPEN(UNIT=2,NAME='TEST1.DAT',TYPE='OLD',READONLY)
5       OPEN(UNIT=2,FILE='INTEST',STATUS='OLD')
C       READ(2,910)NAME                 !TITLE UP TO 30 CHARACTERS
        READ(2,910)NAME
C       READ(2,*)MODE                   !0-ACM, 1-LRI RADAR CONSTANTS
        READ(2,*)MODE
C       READ(2,*)NDATA                  !NOT USED
        READ(2,*)NDATA
C       READ(2,*)DT                     !TIME STEP
        READ(2,*)DT
C       READ(2,*)NTSOUT                 !NO. TIME STEPS/OUTPUT
        READ(2,*)NTSOUT
C       READ(2,*)TINL                   !INITIAL TIME
        READ(2,*)TINL
C       READ(2,*)TFNL                   !FINAL TIME
        READ(2,*)TFNL
C       READ(2,*)RATE                   !ROLL RATE (RAD/SEC)
        READ(2,*)RATE
C       READ(2,*)ALPHA,BETA             !EXPONENTIAL CONSTANTS FOR ROLL
        READ(2,*)ALPHA,BETA
C       READ(2,*)TR1,TF1                !FORWARD ROLL START,STOP TIMES
        READ(2,*)TR1,TF1
C       READ(2,*)TR2,TF2                !REVERSE ROLL START,STOP TIMES
        READ(2,*)TR2,TF2
C       READ(2,*)ANGLE                  !AZIMUTH ANGLE TO TARGET (RAD)
        READ(2,*)ANGLE
C       READ(2,*)W                      !NOT USED
        READ(2,*)W
C       READ(2,*)ASPECT                 !ASPECT ANGLE (RAD)
        READ(2,*)ASPECT
C       READ(2,*)RANGE                  !TARGET RANGE (FT)
        READ(2,*)RANGE
C       READ(2,*)TAS,TAST               !TRUE AIR SPEED-FIGHTER,TARGET
        READ(2,*)TAS,TAST
C       READ(2,*)TRATE                  !TURN RATE (NEGATIVE-AUTO)
        READ(2,*)TRATE
C       READ(2,*)TR3,TF3                !RIGHT TURN START,STOP TIMES
```

```
        READ(2,*)TR3,TF3
C       READ(2,*)TR4,TF4                        !LEFT TURN START,STOP TIMES
        READ(2,*)TR4,TF4
C       READ(2,*)SN1,SN2,SN3,FR                 !NOISE FACTORS
        READ(2,*)SN1,SN2,SN3,FR
C       READ(2,*)IFIL                           !NOT USED
*****ADDED CODE+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     IFIL IS USED TO SELECT FORMATTED OR UNFORMATTED OUTPUT FOR SOFE
C     IFIL=1 RESULTS IN FORMATTED OUTPUT
*****END OF ADDED CODE**************************************************
        READ(2,*)IFIL
C       READ(2,*)HORT1,HORT2                    !TARGET HORIZONTAL TURN TIMES
        READ(2,*)HORT1,HORT2
C       READ(2,*)HORTG,DELTA                    !TARGET GEES AND EXPONENTIAL
        READ(2,*)HORTG,DELTA
C
C****** INITIALIZATION *******
C
        WRITE(5,940) NAME
        IF(MODE.EQ.0)WRITE(5,*)'        MODE = ACM'
        IF(MODE.EQ.1)WRITE(5,*)'        MODE = LRI'
C
C       INTERNAL MISSION DATA GENERATION
C       CALCULATE LOOP CONSTANTS BASED ON TIME AND OUTPUT INDICATORS
C       AND PRINT OUT
C
320     WRITE(5,*)' '
C       T=0.                                    !INITIAL TIME
        T=0.
C       DT2=DT/2.                               !HALF TIME STEP
        DT2=DT/2.
        NT=(TFNL-TINL)/(DT*NTSOUT)+1.0
C       NNT=TFNL/(DT*NTSOUT)+1.0                !TOTAL NUMBER OF LOOPS
        NNT=TFNL/(DT*NTSOUT)+1.0
C       NDIFF=NNT-NT                            !NUMBER OF LOOPS BEFORE FIRST OU
        NDIFF=NNT-NT
        WRITE(5,960) DT,ANGLE,RATE,TRATE,ALPHA,BETA,TR1,TF1,TR2,
     1   TF2,TR3,TF3,TR4,TF4,HORT1,HORT2,HORTG,DELTA
C
C       INITIALIZE AIRCRAFT MODEL
C
        IAIR=0
        CALL AIR
C
C       INITIALIZE RADAR MODEL
C
        IRADAR=0
        CALL RADAR
        WRITE(5,980)RANGE,ASPECT,TAS,TAST,SN1,SN2,SN3,FR
C
C       INITIALIZE FILTER ROUTINES
C
        IFILT=0
        CALL FILTER
```

```
C
C****** SET UP OUTPUT FILE *******
C
        CLOSE(UNIT=2)
*++++ADDED CODE+++++++++++++++++++++++++++++++++++++++++++++++++
C       IF NE=3, THEN SOFE OUTPUT, OTHERWISE BASELINE OUTPUT
        IF(NOUT .NE. 3) THEN
*****END OF ADDED CODE++++++++++++++++++++++++++++++++++++++++++
C       OPEN(UNIT=2,NAME='OUTPUT.DAT',TYPE='NEW')
        OPEN(UNIT=2,FILE='OUTDAT',STATUS='NEW')
        WRITE(2,900)
        WRITE(2,940) NAME
        IF(MODE.NE.1)WRITE(2,*)'         MODE = ACM'
        IF(MODE.EQ.1)WRITE(2,*)'         MODE = LRI'
        WRITE(2,*)' '
        WRITE(2,960)DT,ANGLE,RATE,TRATE,ALPHA,BETA,TR1,
     1   TF1,TR2,TF2,TR3,TF3,TR4,TF4,HORT1,HORT2,HORTG,DELTA
        WRITE(2,980)RANGE,ASPECT,TAS,TAST,SN1,SN2,SN3,FR
*****ADDED CODE*******************************************************
        ENDIF
C       IF IFIL IS 1, SOFE OUTPUT IS FORMATTED, OTHERWISE UNFORMATTED
        IF (NOUT .EQ. 3 .AND. IFIL .EQ. 1) THEN
        OPEN(UNIT=3,FILE='OUTDAT',STATUS='NEW')
        TITLE3='*****APQ-120 RADAR MODEL*****'
        WRITE(3,*)TITLE3
        WRITE(3,940)NAME
        IF(MODE.NE.1)THEN
          MODE3='MODE = ACM'
          WRITE(3,*)MODE3
        ENDIF
        IF(MODE.EQ.1)THEN
          MODE3='MODE = LRI'
          WRITE(3,*)MODE3
        ENDIF
        ENDIF
        IF (NOUT .EQ. 3 .AND. IFIL .NE. 1) THEN
        OPEN(UNIT=3,FILE='OUTDAT',STATUS='NEW',FORM='UNFORMATTED')
        TITLE3='*****APQ-120 RADAR MODEL*****'
        WRITE(3)TITLE3
        WRITE(3)NAME
        IF(MODE.NE.1)THEN
          MODE3='MODE = ACM'
          WRITE(3)MODE3
        ENDIF
        IF(MODE.EQ.1)THEN
          MODE3='MODE = LRI'
          WRITE(3)MODE3
        ENDIF
        ENDIF
****END OF ADDED CODE+++++++++++++++++++++++++++++++++++++++++
C
C       WRITE HEADINGS FOR APPROPRIATE OUTPUT FORMAT
C
C
```

```
C       GO TO (400,410,420),NOUT+1
        GO TO (400,410,420,490),NOUT+1
400     WRITE(5,1000)
        WRITE(2,1000)
        GO TO 490
410     WRITE(5,1010)
        WRITE(2,1010)
        GO TO 490
420     WRITE(5,1020)
        WRITE(2,1020)
C
C******CALCULATIONS AND OUTPUT LOOP******
C
490     DO 750 L=1,NNT
*****ADDED CODE TO INITIALIZE WK1,WJ1,X1,X4,X7***********************
       IF (L .EQ. 1) THEN
          WK1=WK
          WJ1=WJ
          IF (X1(2,1) .LT. 0.)THEN
          XS7=SQRT((RANGE**2*(TAN(X1(2,1)))**2)/(1.+(TAN(X1(2,1)))**2))
          ENDIF
          IF (X1(2,1) .GE. 0.) THEN
          XS7=-SQRT((RANGE**2*(TAN(X1(2,1)))**2)/(1.+(TAN(X1(2,1)))**2))
          ENDIF
          XS1=SQRT((RANGE**2 - XS7**2)/(1. + (TAN(X1(1,1)))**2))
          XS4=XS1*TAN(X1(1,1))
       ENDIF
*****END OF ADDED CODE*********************************************
C                                       DETERMINE OUTPUT
           IF(L.LE.NDIFF) GO TO 575
C                                     INDICATE ROLL
           NPHIDT=0
C                                      RATE 90 PERCENT
           IF(ABS(PHIDT).GE..9*RATE)NPHIDT=1
C          GO TO (500,510,520),NOUT+1
           GO TO (500,510,520,530),NOUT+1
500        WRITE(5,2000)T,NPHIDT,AZ,EL,WK,WJ,TURND,PHIDT,PHI
           WRITE(2,2000)T,NPHIDT,AZ,EL,WK,WJ,TURND,PHIDT,PHI
           GO TO 575
510        WRITE(5,2010)T,NPHIDT,VF(1),VF(2),VF(3),
     1       VT(1),VT(2),VT(3),TGEES
           WRITE(2,2010)T,NPHIDT,VF(1),VF(2),VF(3),
     1      VT(1),VT(2),VT(3),TGEES
           GO TO 575
520        WRITE(5,2020)T,AZ,EL,WK,WJ,TURND,PHIDT,PHI,
     1      VF(1),VF(2),VF(3),VT(1),VT(2),VT(3),TGEES
           WRITE(2,2020)T,AZ,EL,WK,WJ,TURND,PHIDT,PHI,
     1      VF(1),VF(2),VF(3),VT(1),VT(2),VT(3),TGEES
*****ADDED CODE*TO*WRITE*SOFE*OUTPUT*TO*A*FILE*********************
        GO TO 575
530     IF (IFIL.EQ.1)THEN
        WRITE(3,2030)T,RANGE,RDOT,AZ,EL,WK1,WJ1,XS1,VTIJKO(1),
     1   ATIJKO(1),XS4,VTIJKO(2),ATIJKO(2),XS7,VTIJKO(3),ATIJKO(3),
     2   VT1(1),VT1(2),VT1(3),TURN,PHI
```

```
      CALL ROTATE(6,X1(1,1),VTIJKO,VTIJKAZ)
      CALL ROTATE(5,X1(2,1),VTIJKAZ,VTIJK)
      WKTRUE=(VTIJK(2)-VF(2))/RANGE
      WJTRUE=(VF(3)-VTIJK(3))/RANGE
      WRITE(3,*)'TRUE AZ = ',X1(1,1),' AZ = ',AZ,' TRUE EL = ',X1(2,1),
     1 ' EL = ',EL
      WRITE(3,*)'TRUE AZDOT = ',WKTRUE,' AZDOT = ',WK1,
     1 ' TRUE ELDOT = ',WJTRUE,' ELDOT = ',WJ1
      WRITE(3,*)'ERRORS IN DEGREES: AZ, EL, AZDOT, ELDOT'
      RTOD=57.29577951
      WRITE(3,*) (X1(1,1)-AZ)*RTOD,'  ',(X1(2,1)-EL)*RTOD,'  ',
     1 (WKTRUE-WK1)*RTOD,'  ',(WJTRUE-WJ1)*RTOD
      WRITE(3,*)
        ENDIF
        IF (IFIL .NE. 1) THEN
        WRITE(3)T,RANGE,RDOT,AZ,EL,WK1,WJ1,XS1,VTIJKO(1),
     1   ATIJKO(1),XS4,VTIJKO(2),ATIJKO(2),XS7,VTIJKO(3),ATIJKO(3),
     2  VT1(1),VT1(2),VT1(3),TURN,PHI
        ENDIF
*****END OF ADDED CODE**************************************************
575        DO 700 J=1,NTSOUT
                CALL RADAR
                CALL FILTER
*****ADDED CODE FOR SOFE INPUT****************************************
C       CALCULATE OUTPUT VALUES FOR WK1, WJ1
        WK1=WK
        WJ1=WJ
C       CALCULATE POSITION XS VALUES (EL AND AZ ARE IN RADAR REFERENCE)
C       NO ROTATIONS ARE REQUIRED
        IF (X1(2,1) .LT. 0.)THEN
        XS7=SQRT((RANGE**2*(TAN(X1(2,1)))**2)/(1.+(TAN(X1(2,1)))**2))
        ENDIF
        IF (X1(2,1) .GE. 0.)THEN
        XS7=-SQRT((RANGE**2*(TAN(X1(2,1)))**2)/(1.+(TAN(X1(2,1)))**2))
        ENDIF
        XS1=SQRT((RANGE**2 - XS7**2)/(1.+ (TAN(X1(1,1)))**2))
        ENDIF
        XS4=XS1*TAN(X1(1,1))
*****END OF ADDED CODE**************************************************
700        CONTINUE
750     CONTINUE
800     STOP
C
900     FORMAT(/,8X,'***** APQ-120 RADAR MODEL--RADAR.FOR *****',
     1   /,14X,'VERSION 1',//)
C910    FORMAT(//,30A2)
910     FORMAT(//,A)
C940    FORMAT(8X,30A2)
940     FORMAT(8X,A)
950     FORMAT(////,7X,F8.5,7X,F7.4,6X,F7.4,9X,F7.4,/,10X,2F8.5,10X,
     1   2F8.5,5X,F7.1,/,6X,16I1,7X,F8.1,7X,F7.1,6X,F7.4,/)
C
960     FORMAT(10X,'TIME STEP = ',F6.4,5X,'ANGLE = ',F8.5,
     1    5X,'RATE = ',F8.5,/,10X,'TRATE = ',F8.5,5X,'ALPHA = ',F8.2,
     2    5X,'BETA = ',F8.2,/,10X,'TR1 = ',F7.3,3X,'TF1 = ',F7.3,3X,
     3    'TR2 = ',F7.3,3X,'TF2 = ',F7.3,/,10X,'TR3 = ',F7.3,3X,
     4    'TF3 = ',F7.3,3X,'TR4 = ',F7.3,3X,'TF4 = ',F7.3,/,
     5    10X,'HORT1 = ',F7.3,3X,'HORT2 = ',F7.3,3X,'HORTG = ',F5.2,
     6    3X,'DELTA = ',F8.2)
980     FORMAT(10X,'RANGE = ',F8.0,3X,'ASPECT = ',F8.5,3X,'TAS = ',
     1     F7.1,3X,'TAST = ',F7.1,/,10X,'SN1 = ',F6.4,3X,'SN2 = ',F6.4,
     2     3X,'SN3 = ',F6.4,3X,'FR = ',F5.2)
990     FORMAT(7X,F8.5,7X,F7.4,6X,F7.4,9X,F7.4,/,10X,2F8.5,10X,2F8.5,
     1   5X,F7.1,/,6X,16I1,7X,F8.1,7X,F7.1,6X,F7.4,/)
1000    FORMAT(/,2X,'TIME',1X,'ROLL',4X,'ANTAZ',5X,'ANTEL',7X,'WK',
     1    8X,'WJ',6X,'TURND',5X,'PHIDT',5X,'PHI',/)
1010    FORMAT(/,2X,'TIME',2X,'ROLL',4X,'VFI',7X,'VFJ',7X,'VFK',
     1    7X,'VTI',7X,'VTJ',7X,'VTK',5X,'TGEES',/)
1020    FORMAT(/,2X,'TIME',5X,'ANTAZ',4X,'ANTEL',6X,'WK',
     1    7X,'WJ',5X,'TURND',4X,'PHIDT',4X,'PHI',
     2    6X,'VFI',5X,'VFJ',5X,'VFK',5X,'VTI',5X,'VTJ',5X,'VTK',
     3    4X,'TGEES',/)
2000    FORMAT(1X,F6.3,2X,I1,7(2X,F8.5))
2010    FORMAT(1X,F6.3,2X,I1,6(3X,F7.1),3X,F5.2)
2020    FORMAT(1X,F6.3,2X,7(F8.5,1X),6(F7.1,1X),2X,F5.2)
*****ADDED SOFE OUTPUT FORMAT STATEMENT********************************
2030    FORMAT(F5.2,2X,/,6(F14.6,2X),/,3(F14.6,2X),/,3(F14.6,2X),/,
     1          3(F14.6,2X),/,5(F14.6,2X))
        END
C
C**********************************************************************
C
        SUBROUTINE FILTER
C
C       DESCRIPTION:  THIS PROGRAM CALCULATES RAW VALUES OF
C       THE FIGHTER AND TARGET VELOCITY FROM THE INFORMATION
C       RECEIVED FROM THE RADAR AND AIR MODELS.  AN OPTIONAL
C       NOISE GENERATOR AND LOWPASS FILTER IS INCLUDED.
*****ADDED CODE TO CALCULATE TARGET ACCELERATIONS IN VARIOUS FRAMES***
C
        COMMON/MAIN/TINL,TFNL,NNT,NDIFF,NTSOUT,
     1     NDATA,NOUT
        COMMON/STEP/T,DT,DT2
        COMMON/RADAR/Y(2,5),Y1(2,5),Y2(2,5),X1(2,2),OUT(2,2),
     1     A,B,C,D,E,GA,KA,K,COEF1,COEF2,COEF3,COEF4,COEF5,COEF6,
     2     XX(3),YY(3),MODE,IRADAR
        COMMON/AIR/RATE,PHI,PHIDT,PPHI,PITCH,PITCHD,PPITCH,TRATE,
     1     TURN,TURND,PTURN,TR1,TR2,TR3,TR4,TF1,TF2,TF3,TF4,ANGLE,
     2     ASPECT,RANGE,RDOT,RANGEP,AOA,TAS,TAST,ALPHA,BETA,ALPHA1,
     3     ALPBET,BETA1,XFDIST,YFDIST,XTDIST,YTDIST,XTCON,YTCON,
     4     HORT1,HORT2,HORTG,TGEES,DELTA,IAIR,TANG
        COMMON/FILTER/VT(3),VF(3),VR(3),NFIL,IFILT,AZ,EL,WK,WJ,ROLL,
     1     YAW,YNAZ,YNEL,YNWK,YNWJ,YNROLL,YNYAW,FR,SN1,SN2,SN3,CONLP1,
     2     CONLP2,PI,I1,I2,U1,U2,V1,V2,VT1(3),VT2(3),
     3     AT(3),VTIJKO(3),ATIJKO(3)
        COMMON/RK4SAV/SAV(2,5)
*****ADDED DIMENSION VARIABLES FOR LOCAL ARRAYS***********************
      DIMENSION ATXYZ(3),ATLMM(3),ATIJKAZ(3),ATNED(3),VTNED(3),
     1          VTXYZ(3)
        LOGICAL NFIL
        REAL K,KA
C
        IF(IFILT.EQ.1)GO TO 10
C
C       *** FILTER INITIALIZATION ***
C
C       YNAZ=OUT(1,2)                           !ANTENNA AZIMUTH HISTORY
5       YNAZ=OUT(1,2)
C       YNEL=OUT(2,2)                           !ANTENNA ELEVATION HISTO
        YNEL=OUT(2,2)
C       YNWK=OUT(1,1)                           !RATE GYRO AZ HISTORY
        YNWK=OUT(1,1)
C       YNWJ=OUT(2,1)                           !RATE GYRO EL HISTORY
        YNWJ=OUT(2,1)
C       YNROLL=PHI                              !AIRCRAFT ROLL HISTORY
        YNROLL=PHI
C       YNYAW=TURN                              !AIRCRAFT HEADING HISTOR
        YNYAW=TURN
C       PI=3.1415927                            !DEFINE PI
        PI=3.1415926536
C       CONLP1=FR*DT/(1.+FR*DT)                 !LOWPASS FILTER CONSTANT
        CONLP1=FR*DT/(1.+FR*DT)
C       CONLP2=1./(1.+FR*DT)                    !LOWPASS FILTER CONSTANT
        CONLP2=1./(1.+FR*DT)
C       NFIL=.TRUE.                             !NOISE GENERATOR FLAG
        NFIL=.FALSE.
C       I1=5                                    !RANDOM NUMBER SEED
        I1=5
C       I2=6                                    !RANDOM NUMBER SEED
        I2=6
C       IFILT=1                                 !FILTER INITIALIZATION
        IFILT=1
C
*****ADDED CODE TO AVOID NOISE ADDITION(NOISE ADDED IN SOFE) *******
      U1=0.
      U2=0.
      V1=0.
      V2=0.
*****END OF ADDED CODE ***********************************************
C       *** FILTER PROCESSING LOOP ***
C
C       NOISE GENERATOR TO ADD NOISE TO AZ, EL, WK, WJ, ROLL, AND YAW
C       (USES RANDOM NUMBER GENERATOR: RAN(X,Y))
C
C10     NFIL=.NOT.NFIL
*****ADDED CODE TO FORCE NFIL=TRUE----AVOIDS RANDOM NUMBER GENERATOR**
10      NFIL=.TRUE.
        IF(NFIL)GO TO 11
C
C       U1=SQRT(-2.*ALOG(RAN(I1,I2)))           !NOISE GENERATOR
        U1=SQRT(-2.*ALOG(RAN(I1,I2)))
        V1=2.*PI*RAN(I1,I2)
        U2=SQRT(-2.*ALOG(RAN(I1,I2)))
        V2=2.*PI*RAN(I1,I2)
C       AZ=OUT(1,2)+U1*COS(V1)*SN1               !ANTENNA AZIMUTH
        AZ=OUT(1,2)+U1*COS(V1)*SN1
C       EL=OUT(2,2)+U1*SIN(V1)*SN1               !ANTENNA ELEVATION
        EL=OUT(2,2)+U1*SIN(V1)*SN1
C       WK=OUT(1,1)+U2*COS(V2)*SN2               !RATE GYRO AZ
        WK=OUT(1,1)+U2*COS(V2)*SN2
C       WJ=OUT(2,1)+U2*SIN(V2)*SN2               !RATE GYRO EL
        WJ=OUT(2,1)+U2*SIN(V2)*SN2
C       ROLL=PHI+U1*COS(V1)*SN3                  !AIRCRAFT ROLL
        ROLL=PHI+U1*COS(V1)*SN3
C       YAW=TURN+U2*COS(V1)*SN3                  !AIRCRAFT HEADING
        YAW=TURN+U2*COS(V1)*SN3
        GO TO 12
C11     AZ=OUT(1,2)+U1*SIN(V1)*SN1               !ANTENNA AZIMUTH
11      AZ=OUT(1,2)+U1*SIN(V1)*SN1
C       EL=OUT(2,2)+U1*COS(V1)*SN1               !ANTENNA ELEVATION
        EL=OUT(2,2)+U1*COS(V1)*SN1
C       WK=OUT(1,1)+U2*SIN(V2)*SN2               !RATE GYRO AZ
        WK=OUT(1,1)+U2*SIN(V2)*SN2
C       WJ=OUT(2,1)+U2*COS(V2)*SN2               !RATE GYRO EL
        WJ=OUT(2,1)+U2*COS(V2)*SN2
C       ROLL=PHI+U1*SIN(V1)*SN3                  !AIRCRAFT AZ
        ROLL=PHI+U1*SIN(V1)*SN3
C       YAW=TURN+U2*SIN(V1)*SN3                  !AIRCRAFT HEADING
        YAW=TURN+U2*SIN(V1)*SN3
C
C       SINGLE POLE LOWPASS FILTER ON DATA
C
12      IF(FR.EQ.0.)GO TO 14
C       AZ=CONLP1*AZ+CONLP2*YNAZ                 !ANTENNA AZIMUTH
        AZ=CONLP1*AZ+CONLP2*YNAZ
        YNAZ=AZ
C       EL=CONLP1*EL+CONLP2*YNEL                 !ANTENNA ELEVATION
        EL=CONLP1*EL+CONLP2*YNEL
        YNEL=EL
C       WK=CONLP1*WK+CONLP2*YNWK                 !RATE GYRO AZ
        WK=CONLP1*WK+CONLP2*YNWK
        YNWK=WK
C       WJ=CONLP1*WJ+CONLP2*YNWJ                 !RATE GYRO EL
        WJ=CONLP1*WJ+CONLP2*YNWJ
        YNWJ=WJ
C       ROLL=CONLP1*ROLL+CONLP2*YNROLL           !AIRCRAFT ROLL
        ROLL=CONLP1*ROLL+CONLP2*YNROLL
        YNROLL=ROLL
C       YAW=CONLP1*YAW+CONLP2*YNYAW              !AIRCRAFT PITCH
        YAW=CONLP1*YAW+CONLP2*YNYAW
        YNYAW=YAW
C
C       GENERATE VELOCITIES IN ANTENNA COORDINATES
C*****VT1 IS FIGHTER VELOCITY IN RADAR REFERENCE (IO,JO,KO)***********
C
14      VT1(1)=TAS
```

```
        VT1(2)=0.
        VT1(3)=0.
C*****VF  IS FIGHTER VELOCITY IN LINE-OF-SIGHT
        CALL ROTATE(6,AZ,VT1,VT2)
C       CALL ROTATE(5,EL,VT2,VF)                !FIGHTER VELOCITY
        CALL ROTATE(5,EL,VT2,VF)
        VR(1)=-RDOT
        VR(2)=-RANGE*WK
        VR(3)=RANGE*WJ
C       VT(1)=VF(1)-VR(1)                       !TARGET VELOCITY
C*****VT  IS TARGET VELOCITY IN LINE-OF-SIGHT
        VT(1)=VF(1)-VR(1)
        VT(2)=VF(2)-VR(2)
        VT(3)=VF(3)-VR(3)
******THE FOLLOWING HAS BEEN ADDED FOR SOFE INPUT********************
C     TARGET VELOCITY IN N-E PLANE
      VTNED(1)=XTCON/DT2
      VTNED(2)=YTCON/DT2
      VTNED(3)=0.
C     ROTATE FROM NED TO IJKO FRAME (PITCH=0.)
      CALL ROTATE(6,TURN,VTNED,VTXYZ)
      CALL ROTATE(4,PHI,VTXYZ,VTIJKO)
C     TARGET ACCELERATION IN N-E PLANE
      ATNED(1)=TGEES*COS(PI-TANG)*32.2
      ATNED(2)=TGEES*SIN(PI-TANG)*32.2
      ATNED(3)=0.
C     ROTATE FROM NED TO IJKO FRAME (PITCH ANGLE = 0.)
        CALL ROTATE(6,TURN,ATNED,ATXYZ)
      CALL ROTATE (4,PHI,ATXYZ,ATIJKO)
C     ROTATE FROM IO,JO,KO TO IJK TO GET AT(1),AT(2),AT(3)
      CALL ROTATE (6,AZ,ATIJKO,ATIJKAZ)
      CALL ROTATE (5,EL,ATIJKAZ,AT)
*****END OF ADDED CODE***********************************************
300     RETURN
        END
        SUBROUTINE ROTATE(N,ANG,X,Y)
C
C TITLE
C       SUBROUTINE: FN: ROTATE
C
C VERSION
C       V-CC-001
C
C AUTHOR AND DATE
C       DESIGNER -- R. BEAL
C       CODER    -- R. BEAL    JULY 1980
C
C MODIFICATIONS
C       SPR-NNNACSXXX   VERSION #NN   DD-MMM-YY   NAME
C
C          DESCRIPTION
C
C FUNCTION
C       THIS SUBROUTINE PERFORMS GENERAL AIRCRAFT COORDINATE SYSTEM
```

```
C       VECTOR ROTATION.
C
C LOCAL DATA
C       T
C         TEMPORARY MATRIX FOR ROTATION VALUES.
C       CAN
C         COSINE OF ROTATION ANGLE.
C       SAN
C         SINE OF ROTATION ANGLE.
C
C CALLING SEQUENCE AND CONDITIONS
C       THIS SUBROUTINE CAN BE CALLED BY ANY PROGRAM.  THE CALL STRING
C       IS AS FOLLOWS:
C           N   - INDEX FOR ROTATION TYPE
C           ANG - ANGLE FOR ROTATION  (MUST BE IN RADIANS).
C           X   - INPUT VECTOR.
C           Y   - OUTPUT VECTOR.
C
C SUBROUTINE/FUNCTION SUBPROGRAMS
C       NONE
C
C COMMENTS
C       NONE
C
C       LOCAL DECLARATION STATEMENT(S)
C
        DIMENSION T(3,3),X(3),Y(3)
C
        CAN=COS(ANG)
        SAN=SIN(ANG)
C
C       GO TO (10,20,30,40,50,60), N    !SET UP THE ROTATION MATRIX
        GO TO (10,20,30,40,50,60), N
C       N = 1
C       PHI (ROLL ANGLE) - INVERSE
10      T(1,1)=1.
        T(1,2)=0.
        T(1,3)=0.
        T(2,1)=0.
        T(2,2)=CAN
        T(2,3)=-SAN
        T(3,1)=0.
        T(3,2)=SAN
        T(3,3)=CAN
        GO TO 70
C       N = 2
C       THETA (PITCH ANGLE), LAMBDA EL, -2 DEGREES, ALPHA - INVERSE
20      T(1,1)=CAN
        T(1,2)=0.
        T(1,3)=SAN
        T(2,1)=0.
        T(2,2)=1.
        T(2,3)=0.
        T(3,1)=-SAN
```

```
          T(3,2)=0.
          T(3,3)=CAN
          GO TO 70
C         N = 3
C         LAMBDA AZ, PSI (HEADING ANGLE) - INVERSE
30        T(1,1)=CAN
          T(1,2)=-SAN
          T(1,3)=0.
          T(2,1)=SAN
          T(2,2)=CAN
          T(2,3)=0.
          T(3,1)=0.
          T(3,2)=0.
          T(3,3)=1.
          GO TO 70
C         N = 4
C         PHI (ROLL ANGLE)
40        T(1,1)=1.
          T(1,2)=0.
          T(1,3)=0.
          T(2,1)=0.
          T(2,2)=CAN
          T(2,3)=SAN
          T(3,1)=0.
          T(3,2)=-SAN
          T(3,3)=CAN
          GO TO 70
C         N = 5
C         THETA (PITCH ANGLE), LAMBDA EL, -2 DEGREES, ALPHA
50        T(1,1)=CAN
          T(1,2)=0.
          T(1,3)=-SAN
          T(2,1)=0.
          T(2,2)=1.
          T(2,3)=0.
          T(3,1)=SAN
          T(3,2)=0.
          T(3,3)=CAN
          GO TO 70
C         N = 6
C         LAMBDA AZ, PSI (HEADING ANGLE)
60        T(1,1)=CAN
          T(1,2)=SAN
          T(1,3)=0.
          T(2,1)=-SAN
          T(2,2)=CAN
          T(2,3)=0.
          T(3,1)=0.
C
          T(3,2)=0.
          T(3,3)=1.
C         PERFORM THE ROTATION
70        Y(1)=T(1,1)*X(1)+T(1,2)*X(2)+T(1,3)*X(3)
          Y(2)=T(2,1)*X(1)+T(2,2)*X(2)+T(2,3)*X(3)
```

```
        Y(3)=T(3,1)*X(1)+T(3,2)*X(2)+T(3,3)*X(3)
        RETURN
        END
C
C****** APQ-120 RADAR MODEL--RADAR.FTN *******
C       VERSION 1
C
C       DESCRIPTION:  THESE SUBROUTINES REPRESENTS A COMMON SET
C       FOR RADAR.FTN PROGRAMS.  INCLUDED ARE THE APQ-120 RADAR
C       MODEL, AN AIRCRAFT/TARGET GEOMETRY MODEL AND A SUPPORTING
C       FOURTH ORDER RUNGE-KUTTA ROUTINE.
C
C
        SUBROUTINE RADAR
C
C****** RADAR MODEL *******
C
C       DESCRIPTION:  THIS SUBPROGRAM MODELS THE APQ-120 RADAR FOR BOTH
C       ACM AND LRI MODES.  RADAR CONTAINS AUTOMATIC GAIN CONSTANT
C       CALCULATION AND BORESIGHT ROTATION.
C
        COMMON/STEP/T,DT,DT2
        COMMON/RADAR/Y(2,5),Y1(2,5),Y2(2,5),X1(2,2),OUT(2,2),
     1    A,B,C,D,E,GA,KA,K,COEF1,COEF2,COEF3,COEF4,COEF5,COEF6,
     2    XX(3),YY(3),MODE,IRADAR
        COMMON/AIR/RATE,PHI,PHIDT,PPHI,PITCH,PITCHD,PPITCH,TRATE,
     1    TURN,TURND,PTURN,TR1,TR2,TR3,TR4,TF1,TF2,TF3,TF4,ANGLE,
     2    ASPECT,RANGE,RDOT,RANGEP,AOA,TAS,TAST,ALPHA,BETA,ALPHA1,
     3    ALPBET,BETA1,XFDIST,YFDIST,XTDIST,YTDIST,XTCON,YTCON,
     4    HORT1,HORT2,HORTG,TGEES,DELTA,IAIR,TANG
        COMMON/RK4SAV/SAV(2,5)
        REAL K,KA
C
        IF(IRADAR.EQ.1)GO TO 70
C
C       INITIALIZE RADAR MODEL
C       SET SERVO CONSTANTS FOR SELECTED RADAR MODE
C
        IF(MODE.EQ.1)GO TO 20
        A=.4995
        B=.0495
        GA=17.47
        GO TO 30
20      A=2.008
        B=.1990
        GA=1.092
30      C=.0576
        D=.2576
        E=2.0
        KA=61.31
        K=2.164
C
C       SET UP COEFICIENTS FOR ANTENNA MODEL RUNGE KUTTA APPROX.
C
```

```
        COEF1=-1./B
        COEF2=KA/D
        COEF3=KA*C/D
        COEF4=-1./D
        COEF5=K/2./B
        COEF6=K/2.*A/B
        DO 40 I=1,5
           Y(1,I)=0.
           Y(2,I)=0.
           Y1(1,I)=0.
           Y1(2,I)=0.
40      CONTINUE
C       Y(1,3)=ANGLE                            !ANTENNA AZ (RAD)
        Y(1,3)=ANGLE
C       Y(2,3)=0.                               !ANTENNA EL (RAD)
        Y(2,3)=0.
C       X1(1,1)=(ANGLE-TURN)*COS(PHI)           !AZIMUTH INPUT (RAD)
        X1(1,1)=(ANGLE-TURN)*COS(PHI)
C       X1(2,1)=(ANGLE-TURN)*SIN(PHI)           !ELEVATION INPUT (RAD)
        X1(2,1)=(ANGLE-TURN)*SIN(PHI)
C       X1(1,2)=0.                              !AZ RATE INPUT (RAD/SEC)
        X1(1,2)=0.
C       X1(2,2)=0.                              !EL RATE INPUT (RAD/SEC)
        X1(2,2)=0.
C       OUT(1,1)=Y(1,4)*COS(Y(2,3))+X1(1,2)     !RATE GYRO AZ (RAD/SEC)
        OUT(1,1)=Y(1,4)*COS(Y(2,3))+X1(1,2)
C       OUT(2,1)=Y(2,4)+X1(2,2)                 !RATE GYRO EL (RAD/SEC)
        OUT(2,1)=Y(2,4)+X1(2,2)
C       OUT(1,2)=Y(1,3)                         !ANTENNA AZ OUTPUT (RAD)
        OUT(1,2)=Y(1,3)
C       OUT(2,2)=Y(2,3)                         !ANTENNA EL OUTPUT (RAD)
        OUT(2,2)=Y(2,3)
C       IRADAR=1                                !RADAR MODEL INITIALIZED
        IRADAR=1
        GO TO 90
C
C       RADAR MODEL PROCESSING
C
70      DO 85 N=1,4
        DO 80 I=1,2
           F=1.
           IF(I.EQ.1)F=COS(OUT(2,2))
           Y1(I,1)=Y(I,2)
           Y1(I,2)=COEF1*Y(I,2) + AGC1*GA*X1(I,1) -
     1      AGC1*GA*Y(I,3)
           Y1(I,3)=Y(I,4)
           Y1(I,4)=COEF2*Y(I,5) + COEF3*(COEF4*Y(I,5) -
C
     1      E*F*Y(I,4) + COEF5*Y(I,1) + COEF6*Y(I,2) -
     2      E*X1(I,2))
           Y1(I,5)=COEF4*Y(I,5) - E*F*Y(I,4) + COEF5*Y(I,1) +
     1      COEF6*Y(I,2) - E*X1(I,2)
           CALL RK4(N,5,T,I)
           IF(N.EQ.2.OR.N.EQ.4)GO TO 80
           IF(I.EQ.2)GO TO 73
           T=T-DT2
           GO TO 80
C
C       GENERATE RADAR MODEL INPUTS
C
C  73      IF(I.EQ.2)CALL AIR                          !UPDATE AIR MODEL
73         IF(I.EQ.2)CALL AIR
           X1(1,1)=(ANGLE-TURN)*COS(PHI)
           X1(2,1)=(ANGLE-TURN)*SIN(PHI)
           XX(1)=PHIDT-TURND*SIN(PITCH)
           XX(2)=PITCHD*COS(PHI)+TURND*COS(PITCH)*SIN(PHI)
           XX(3)=-PITCHD*SIN(PHI)+TURND*COS(PITCH)*COS(PHI)
           CALL ROTATE(5,-.0349066,XX,YY)
           CALL ROTATE(6,Y(1,3),YY,XX)
           CALL ROTATE(5,Y(2,3),XX,YY)
           X1(1,2)=YY(3)
           X1(2,2)=YY(2)
80      CONTINUE
85      CONTINUE
C       OUT(1,1)=Y(1,4)*COS(Y(2,3))+X1(1,2)         !RATE GYRO AZ
        OUT(1,1)=Y(1,4)*COS(Y(2,3))+X1(1,2)
C       OUT(2,1)=Y(2,4)+X1(2,2)                     !RATE GYRO EL
        OUT(2,1)=Y(2,4)+X1(2,2)
C 86    OUT(1,2)=Y(1,3)                             !ANTENNA AZ OUTPUT
86      OUT(1,2)=Y(1,3)
C       OUT(2,2)=Y(2,3)                             !ANTENNA EL OUTPUT
        OUT(2,2)=Y(2,3)
C
C      AUTOMATIC GAIN CONSTANT CALCULATION
C
90      B2=(X1(1,1)-OUT(1,2))**2.+(X1(2,1)-OUT(2,2))**2.
        GAMM=SQRT(B2)
        IF(GAMM.GE..34)GO TO 120
        GAMM=GAMM-.02625
        IF(GAMM.LT.0.)GO TO 110
        BTH=27.*GAMM
        G2=SIN(BTH)/BTH
        G2=G2*G2/(1.+.71*BTH*BTH)
        G2=G2*G2
        GO TO 150
110     G2=1.
        GO TO 150
120     G2=0.
150     S=1.278419E9 * G2/(RANGE*RANGE*RANGE*RANGE)
        AGC1=S/(S+6.31E-14)
C
        RETURN
        END
C
C
        SUBROUTINE AIR
C
C****** AIRCRAFT MODEL *******
```

```
C
C       DESCRIPTION:  THIS SUBPROGRAM GENERATES A SIMULATION OF
C       THE AIRCRAFT AND TARGET IN FLIGHT.
C
        COMMON/STEP/T,DT,DT2
        COMMON/RADAR/Y(2,5),Y1(2,5),Y2(2,5),X1(2,2),OUT(2,2),
     1     A,B,C,D,E,GA,KA,K,COEF1,COEF2,COEF3,COEF4,COEF5,COEF6,
     2     XX(3),YY(3),MODE,IRADAR
        COMMON/AIR/RATE,PHI,PHIDT,PPHI,PITCH,PITCHD,PPITCH,TRATE,
     1     TURN,TURND,PTURN,TR1,TR2,TR3,TR4,TF1,TF2,TF3,TF4,ANGLE,
     2     ASPECT,RANGE,RDOT,RANGEP,AOA,TAS,TAST,ALPHA,BETA,ALPHA1,
     3     ALPBET,BETA1,XFDIST,YFDIST,XTDIST,YTDIST,XTCON,YTCON,
     4     HORT1,HORT2,HORTG,TGEES,DELTA,IAIR,TANG
        COMMON/RK4SAV/SAV(2,5)
        REAL K,KA
C
        IF(IAIR.EQ.1)GO TO 20
C
C       INITIALIZE AIRCRAFT MODEL
C
C       PHI=0.                                          !ROLL ANGLE(RAD)
        PHI=0.
C       PHIDT=0.                                        !ROLL RATE(RAD/SEC)
        PHIDT=0.
C       PPHI=0.                                         !PREVIOUS ROLL ANGLE
        PPHI=0.
C       TURN=0.                                         !YAW ANGLE(RAD)
        TURN=0.
C       TURND=0.                                        !YAW RATE(RAD/SEC)
        TURND=0.
C       PTURN=0.                                        !PREVIOUS YAW ANGLE
        PTURN=0.
C       PITCH=0.                                        !AIRCRAFT PITCH (RAD)
        PITCH=0.
C       PITCHD=0.                                       !PITCH RATE (RAD/SEC)
        PITCHD=0.
C       PPITCH=PITCH                                    !PREVIOUS PITCH ANGLE
        PPITCH=PITCH
C       AOA=0.                                          !ANGLE OF ATTACK (RAD)
        AOA=0.
***** ADDED CODE FOR PSEUDO INITIALIZATION *****************
      PHIDT1=0.
      PHIDT2=0.
      PHI1=0.
      PHI2=0.
      TURND1=0.
C
      TURND2=0.
      TURN1=0.
      TURN2=0.
      TANG=0.
*****END OF PSEUDO INITIALIZATION*****************************************
C
C       CALCULATE ROLL AND TURN CONSTANTS
```

```
C
        ALPHA1=1./ALPHA
        ALPBET=ALPHA+BETA
        BETA1=1./ALPBET
C
C       SET UP GEOMETRY (X-AXIS IS IN 0-YAW DIRECTION)
C
C       XFDIST=0.                               !FIGHTER X LOCATION (FT)
        XFDIST=0.
C       YFDIST=0.                               !FIGHTER Y LOCATION (FT)
        YFDIST=0.
C       XTDIST=RANGE*COS(ANGLE)                 !TARGET X LOCATION (FT)
        XTDIST=RANGE*COS(ANGLE)
C       YTDIST=RANGE*SIN(ANGLE)                 !TARGET Y LOCATION (FT)
        YTDIST=RANGE*SIN(ANGLE)
C       RDOT=-(TAST*COS(ASPECT)+TAS*COS(ANGLE)) !INITIAL RANGE RATE (FT/SE
        RDOT=-(TAST*COS(ASPECT)+TAS*COS(ANGLE))
C       RANGEP=RANGE                            !PREVIOUS RANGE VALUE (FT)
        RANGEP=RANGE
C       XTCON=COS(ANGLE-3.141593+ASPECT)*TAST*
C    1     DT2                                  !TARGET X VELOCITY (FT/SEC
        XTCON=COS(ANGLE-3.141593+ASPECT)*TAST*DT2
C       YTCON=SIN(ANGLE-3.141593+ASPECT)*TAST*
C    1     DT2                                  !TARGET Y VELOCITY (FT/SEC
        YTCON=SIN(ANGLE-3.141593+ASPECT)*TAST*DT2
        TGEES=0.
        IAIR=1
        GO TO 100
C
C       AIRCRAFT MODEL PROCESSING
C       GENERATE ROLL INPUTS
C
20      IF(T.GE.TR1)PHIDT1=RATE/(ALPHA1-BETA1)*(-EXP
     1   (-ALPHA*(T-TR1))*ALPHA1+EXP(-ALPBET*(T-TR1))
     2   *BETA1+ALPHA1-BETA1)
        IF(T.GE.TR2)PHIDT2=-RATE/(ALPHA1-BETA1)*(-EXP
     1   (-ALPHA*(T-TR2))*ALPHA1+EXP(-ALPBET*(T-TR2))
     2   *BETA1+ALPHA1-BETA1)
        IF(T.GE.TF1)PHIDT1=PHIDT1-RATE/(ALPHA1-BETA1)*
     1   (-EXP(-ALPHA*(T-TF1))*ALPHA1+EXP(-ALPBET*(T-TF1))
     2   *BETA1+ALPHA1-BETA1)
        IF(T.GE.TF2)PHIDT2=PHIDT2+RATE/(ALPHA1-BETA1)*
     1   (-EXP(-ALPHA*(T-TF2))*ALPHA1+EXP(-ALPBET*(T-TF2))
     2   *BETA1+ALPHA1-BETA1)
        PHIDT=PHIDT1+PHIDT2
        IF(T.GE.TR1)PHI1=RATE/(ALPHA1-BETA1)*(EXP(-ALPHA*
     1   (T-TR1))*ALPHA1*ALPHA1-EXP(-ALPBET*(T-TR1))*
     2   BETA1*BETA1+(T-TR1)*ALPHA1-(T-TR1)*BETA1-ALPHA1
     3   *ALPHA1+BETA1*BETA1)
        IF(T.GE.TR2)PHI2=-RATE/(ALPHA1-BETA1)*(EXP(-ALPHA*
     1   (T-TR2))*ALPHA1*ALPHA1-EXP(-ALPBET*(T-TR2))*
     2   BETA1*BETA1+(T-TR2)*ALPHA1-(T-TR2)*BETA1-ALPHA1
     3   *ALPHA1+BETA1*BETA1)
        IF(T.GE.TF1)PHI1=PHI1-RATE/(ALPHA1-BETA1)*(EXP
     1  (-ALPHA*(T-TF1))*ALPHA1*ALPHA1-EXP(-ALPBET*(T-
     2  TF1))*BETA1*BETA1+(T-TF1)*ALPHA1-(T-TF1)*BETA1
     3  -ALPHA1*ALPHA1+BETA1*BETA1)
       IF(T.GE.TF2)PHI2=PHI2+RATE/(ALPHA1-BETA1)*(EXP
     1  (-ALPHA*(T-TF2))*ALPHA1*ALPHA1-EXP(-ALPBET*(T-
     2  TF2))*BETA1*BETA1+(T-TF2)*ALPHA1-(T-TF2)*BETA1
     3  -ALPHA1*ALPHA1+BETA1*BETA1)
       PHI=PHI1+PHI2
C
C      GENERATE TURN INPUTS
C
       TURND=32.2*TAN(PHI)/TAS
       TURN=TURN+TURND*DT2
       IF(TRATE.LT.0.)GO TO 75
       IF(T.GE.TR3)TURND1=TRATE/(ALPHA1-BETA1)*(-EXP
     1  (-ALPHA*(T-TR3))*ALPHA1+EXP(-ALPBET*(T-TR3))*
     2  BETA1+ALPHA1-BETA1)
       IF(T.GE.TR4)TURND2=-TRATE/(ALPHA1-BETA1)*(-EXP
     1  (-ALPHA*(T-TR4))*ALPHA1+EXP(-ALPBET*(T-TR4))*
     2  BETA1+ALPHA1-BETA1)
       IF(T.GE.TF3)TURND1=TURND1-TRATE/(ALPHA1-BETA1)*
     1  (-EXP(-ALPHA*(T-TF3))*ALPHA1+EXP(-ALPBET*(T-TF3))
     2  *BETA1+ALPHA1-BETA1)
       IF(T.GE.TF4)TURND2=TURND2+TRATE/(ALPHA1-BETA1)*
     1  (-EXP(-ALPHA*(T-TF4))*ALPHA1+EXP(-ALPBET*(T-TF4))
     2  *BETA1+ALPHA1-BETA1)
       TURND=TURND1+TURND2
       IF(T.GE.TR3)TURN1=TRATE/(ALPHA1-BETA1)*(EXP
     1  (-ALPHA*(T-TR3))*ALPHA1*ALPHA1-EXP(-ALPBET*
     2  (T-TR3))*BETA1*BETA1+(T-TR3)*ALPHA1-
     3  (T-TR3)*BETA1-ALPHA1*ALPHA1+BETA1*BETA1)
       IF(T.GE.TR4)TURN2=-TRATE/(ALPHA1-BETA1)*(EXP
     1  (-ALPHA*(T-TR4))*ALPHA1*ALPHA1-EXP(-ALPBET*
     2  (T-TR4))*BETA1*BETA1+(T-TR4)*ALPHA1-
     3  (T-TR4)*BETA1-ALPHA1*ALPHA1+BETA1*BETA1)
       IF(T.GE.TF3)TURN1=TURN1-TRATE/(ALPHA1-BETA1)
     1  *(EXP(-ALPHA*(T-TF3))*ALPHA1*ALPHA1-EXP
     2  (-ALPBET*(T-TF3))*BETA1*BETA1+(T-TF3)*
     3  ALPHA1-(T-TF3)*BETA1-ALPHA1*ALPHA1+BETA1*BETA1)
       IF(T.GE.TF4)TURN2=TURN2+TRATE/(ALPHA1-BETA1)
     1  *(EXP(-ALPHA*(T-TF4))*ALPHA1*ALPHA1-EXP
     2  (-ALPBET*(T-TF4))*BETA1*BETA1+(T-TF4)*
     3  ALPHA1-(T-TF4)*BETA1-ALPHA1*ALPHA1+BETA1*BETA1)
       TURN=TURN1+TURN2
C
C
C      TARGET MANUEVER
C
75     IF(HORTG.EQ.0.)GO TO 80
       TGEES=0.
       IF(T.GE.HORT1)TGEES=HORTG*(1.-EXP(-DELTA*(T-HORT1)))
       IF(T.GE.HORT2)TGEES=HORTG*EXP(-DELTA*(T-HORT2))
       TANG=ATAN2(XTCON,YTCON)
       XTCON=XTCON-32.2*TGEES*COS(TANG)*DT2*DT2
```

```
        YTCON=YTCON+32.2*TGEES*SIN(TANG)*DT2*DT2
C
C       UPDATE GEOMETRY
C
80      XFDIST=XFDIST+COS(TURN)*TAS*DT2
        YFDIST=YFDIST+SIN(TURN)*TAS*DT2
        XTDIST=XTDIST+XTCON
        YTDIST=YTDIST+YTCON
        ANGLE=ATAN((YTDIST-YFDIST)/(XTDIST-XFDIST))
        RANGE=SQRT((XTDIST-XFDIST)**2.+(YTDIST-YFDIST)**2.)
        RDOT=(RANGE-RANGEP)/DT2
        RANGEP=RANGE
C
100     RETURN
        END
C
C
        SUBROUTINE RK4(M,N,TT,II)
C
C****** FOURTH ORDER RUNGE-KUTTA ROUTINE *******
C
        COMMON/STEP/T,DT,DT2
        COMMON/RADAR/Y(2,5),Y1(2,5),Y2(2,5),X1(2,2),OUT(2,2),
     1     A,B,C,D,E,GA,KA,K,COEF1,COEF2,COEF3,COEF4,COEF5,COEF6,
     2     XX(3),YY(3),MODE,IRADAR
        COMMON/AIR/RATE,PHI,PHIDT,PPHI,PITCH,PITCHD,PPITCH,TRATE,
     1     TURN,TURND,PTURN,TR1,TR2,TR3,TR4,TF1,TF2,TF3,TF4,ANGLE,
     2     ASPECT,RANGE,RDOT,RANGEP,AOA,TAS,TAST,ALPHA,BETA,ALPHA1,
     3     ALPBET,BETA1,XFDIST,YFDIST,XTDIST,YTDIST,XTCON,YTCON,
     4     HORT1,HORT2,HORTG,TGEES,DELTA,IAIR,TANG
        COMMON/RK4SAV/SAV(2,5)
        REAL K,KA
C
        GO TO (200,210,220,230),M
200     DO 201 I=1,N
        Y2(II,I)=Y(II,I)
        CO=DT2*Y1(II,I)
        Y(II,I)=Y2(II,I)+CO
201     SAV(II,I)=CO+CO
        TT=TT+DT2
        RETURN
210     DO 211 I=1,N
        CO=DT2*Y1(II,I)
        Y(II,I)=Y2(II,I)+CO
211     SAV(II,I)=SAV(II,I)+4.0*CO
        RETURN
220     DO 221 I=1,N
        CO=DT*Y1(II,I)
        Y(II,I)=Y2(II,I)+CO
221     SAV(II,I)=SAV(II,I)+CO+CO
        TT=TT+DT2
        RETURN
230     DO 231 I=1,N
231     Y(II,I)=Y2(II,I)+(SAV(II,I)+DT*Y1(II,I))/6.0
        RETURN
        END
```

```
************************************************************
******* TEST DEFINITION FILE FOR RADAR.FTN PROGRAMS *******
BEAM ANGLE 0 DEG, LAG
0               ;MODE (0-ACM,1-LRI)
0               ;NDATA-INPUT DATA (0-INTERNAL, 1-EXTERNAL)
.02             ;DT-TIME STEP
1               ;NTSOUT-NO. TIME STEPS/OUTPUT
0.              ;TINL-INITIAL TIME
15.30           ;TFNL-FINAL TIME
1.              ;RATE-ROLL RATE
5.,10.          ;ALPHA,BETA-EXPONENTIAL CONSTANTS FOR ROLL AND TURN
3.,4.           ;TR1,TF1-FORWARD START,STOP TIMES
5.,6.           ;TR2,TF2-REVERSE START,STOP TIMES
.785398163      ;ANGLE-INITIAL AZIMUTH ANGLE TO TARGET
3.5             ;FILTER CONSTANT
.785398163      ;ASPECT ANGLE OR RDOT-RANGE RATE
40000.          ;RANGE TO TARGET
800.,800.       ;TRUE AIR SPEED-FIGHTER,TARGET
-1.             ;TURN RATE (NEGATIVE FOR AUTO RATE)
0.,0.           ;TR3,TF3-RIGHT TURN START,STOP TIMES (MANUAL)
0.,0.           ;TR4,TF4-LEFT TURN START,STOP TIMES (MANUAL)
0.,.0001,0.,18.85       ;SN1,SN2,SN3,FR
0               ;IF 1,FORMATTED OUTPUT,OTHERWISE UNFORMATTED
1.,9.           ;HORT1,HORT2-HORIZONTAL TURN START,STOP TIMES
3.,5.           ;HORTG(+AWAY,-TOWARDS),DELTA-TARGET GEES, EXPONENTIAL
--EOR--
```

```
***********************************************************
******* TEST DEFINITION FILE FOR RADAR.FTN PROGRAMS *******
TAIL CHASE TRAJ
0               ;MODE (0-ACM,1-LRI)
0               ;NDATA-INPUT DATA (0-INTERNAL, 1-EXTERNAL)
.02             ;DT-TIME STEP
1               ;NTSOUT-NO. TIME STEPS/OUTPUT
0.              ;TINL-INITIAL TIME
15.30           ;TFNL-FINAL TIME
1.              ;RATE-ROLL RATE
5.,10.          ;ALPHA,BETA-EXPONENTIAL CONSTANTS FOR ROLL AND TURN
5.,6.           ;TR1,TF1-FORWARD START,STOP TIMES
16.,16.         ;TR2,TF2-REVERSE START,STOP TIMES
-.087266        ;ANGLE-INITIAL AZIMUTH ANGLE TO TARGET
3.5             ;FILTER CONSTANT
3.228859        ;ASPECT ANGLE OR RDOT-RANGE RATE
10000.          ;RANGE TO TARGET
800.,800.       ;TRUE AIR SPEED-FIGHTER,TARGET
-1.             ;TURN RATE (NEGATIVE FOR AUTO RATE)
0.,0.           ;TR3,TF3-RIGHT TURN START,STOP TIMES (MANUAL)
0.,0.           ;TR4,TF4-LEFT TURN START,STOP TIMES (MANUAL)
0.,.0001,0.,18.85       ;SN1,SN2,SN3,FR
1               ;IF 1,FORMATTED OUTPUT,OTHERWISE UNFORMATTED
1.,16.          ;HORT1,HORT2-HORIZONTAL TURN START,STOP TIMES
3.,5.           ;HORTG(+AWAY,-TOWARDS),DELTA-TARGET GEES, EXPONENTIAL
--EOR--
```

Appendix C

SOFE User-Written Programs

```
*DECK FQGEN
      SUBROUTINE  FQGEN (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, NZF, NZQ
     +                          F, QF)
C
C +++ =FQGEN=, A USER-WRITTEN SUBROUTINE FOR SOFE.
C +++ CALLED ON DEMAND OF THE INTEGRATOR.  SUPPLIES THE NONZERO
C +++ ELEMENTS OF THE MATRICES F AND QF FOR PROPAGATION OF THE
C +++ COVARIANCE MATRIX PF.  THERE ARE NOT ANY TIME VARYING VALUES
C +++ OF MATRIX F BECAUSE OF LINEAR DYNAMICS -- SO RETURN.
C
      COMMON/QF/QFIN(3)
C
      DIMENSION  XF(NF), XS(NS), XTRAJ(NXTJ), F(NZF), QF(NZQ)
C
      RETURN
C
C     *********************
C     * FQGEN ENTRY POINT *
C     *********************
C
      ENTRY  FQGENO (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, NZF, NZQ,
     +                      F, QF)
C
      F(1)=1.
      F(2)=1.
      F(3)=-7.
      F(4)=1.
      F(5)=1.
      F(6)=-7.
      F(7)=1.
      F(8)=1.
      F(9)=-7.
      QF(1)=QFIN(1)
      QF(2)=QFIN(2)
      QF(3)=QFIN(3)
      RETURN
      END
*DECK HRZ
      SUBROUTINE HRZ (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, NTR, PF,
     +                        IMEAS, M, H, RF, ZRES)
C
C +++ =HRZ=, A USER-WRITTEN SUBROUTINE FOR SOFE.
C +++ CALLED M TIMES AT DTMEAS INTERVALS.  SUPPLIES SENSITIVITY
C +++ VECTOR H, NOISE VARIANCE RF, AND RESIDUAL ZRES.
C +++ MAKE RF < 0 TO SUPPRESS A MEASUREMENT.
C
      COMMON/RFCOM/RFIN(6)
      COMMON/HHCOM/HH(24)
C
      DIMENSION  XF(NF), XS(NS), XTRAJ(NXTJ), PF(NTR), H(NF),STD(6)
C
C     FIRST,  EVALUATE H FOR ALL MEASUREMENTS
      IF (IMEAS .EQ. 1) THEN
```

```
         SUMSQ = XF(1)**2 + XF(4)**2 + XF(7)**2
         SUMSQ12= SQRT(SUMSQ)
         SUMSQ32= (SUMSQ)**1.5
         SUMSQ2 = (SUMSQ)**2
         PSUM   = XF(1)**2 + XF(4)**2
         PSUM12 = SQRT(PSUM)
         PSUM2  = PSUM**2
         PSUM32 = PSUM**1.5
         ADD1   = (XF(1)*(XF(2)-XTRAJ(16)))+XF(4)*XF(5) + XF(7)*XF(8)
         ADD2   = XF(4)**2 - XF(1)**2
         ADD3   = XF(8)*PSUM - XF(7)*(XF(1)*(XF(2)-XTRAJ(16)) +
     1  XF(4)*XF(5))
         HH(1)  = XF(1)/SUMSQ12
         HH(2)  = XF(4)/SUMSQ12
         HH(3)  = XF(7)/SUMSQ12
         HH(4)  = ((XF(2)-XTRAJ(16))*SUMSQ - XF(1)*ADD1)/SUMSQ32
         HH(5)  = HH(1)
         HH(6)  = (XF(5)*SUMSQ - XF(4)*ADD1)/SUMSQ32
         HH(7)  = HH(2)
         HH(8)  = (XF(8)*SUMSQ - XF(7)*ADD1)/SUMSQ32
         HH(9)  = HH(3)
         HH(10) = -XF(4)/PSUM
         HH(11) = XF(1)/PSUM
         HH(12) = (XF(1)*XF(7))/(SUMSQ*PSUM12)
         HH(13) = (XF(4)*XF(7))/(SUMSQ*PSUM12)
         HH(14) = -PSUM12/SUMSQ
         HH(15) = (XF(5)*ADD2 + 2.*XF(1)*(XF(2)-XTRAJ(16))*XF(4))
     1            /PSUM2
         HH(16) = HH(10)
         HH(17) = ((XF(2)-XTRAJ(16))*ADD2 - 2.*XF(1)*XF(4)*XF(5))
     1            /PSUM2
         HH(18) = HH(11)
         HH(19) =-((SUMSQ*PSUM*(2.*XF(1)*XF(8) -
     1            (XF(2)-XTRAJ(16))*XF(7))) - ADD3*
     2            XF(1)*(2.*PSUM + SUMSQ))/(SUMSQ2*PSUM32)
         HH(20) = HH(12)
         HH(21) =-((SUMSQ*PSUM*(2.*XF(4)*XF(8) - XF(5)*XF(7))) - ADD3*
     1            XF(4)*(2.*PSUM + SUMSQ))/(SUMSQ2*PSUM32)
         HH(22) = HH(13)
         HH(23) = ((SUMSQ*PSUM*(XF(1)*(XF(2)-XTRAJ(16)) +
     1            XF(4)*XF(5)) + ADD3*
     2            2.*XF(7)*PSUM))/(SUMSQ2*PSUM32)
         HH(24) = -PSUM/(SUMSQ*PSUM12)
      ENDIF
C
C     NOW GET RF
      RF = RFIN(IMEAS)
C
C     NOW OBTAIN MEASUREMENT RESIDUALS, SEQUENTUALLY.  (ZRES = ZS - ZF)
      DO 5 I=1,NF
         H(I)=0.
    5 CONTINUE
C
```

```
C     RANGE
      IF (IMEAS .EQ. 1) THEN
         ZRES = XTRAJ(1) - SUMSQ12 + GAUSS(0.,STD(1))
         H(1) = HH(1)
         H(4) = HH(2)
         H(7) = HH(3)
      ENDIF
C
C     RANGE RATE
      IF (IMEAS .EQ. 2) THEN
         ZRES = XTRAJ(2) - ADD1/SUMSQ12 + GAUSS(0.,STD(2))
         H(1) = HH(4)
         H(2) = HH(5)
         H(4) = HH(6)
         H(5) = HH(7)
         H(7) = HH(8)
         H(8) = HH(9)
      ENDIF
C
C     AZIMUTH ANGLE
      IF (IMEAS .EQ. 3) THEN
         ZRES = XTRAJ(3) -ATAN(XF(4)/XF(1)) + GAUSS(0.,STD(3))
         H(1) = HH(10)
         H(4) = HH(11)
      ENDIF
C
C     ELEVATION ANGLE
      IF (IMEAS .EQ. 4) THEN
         ZRES = XTRAJ(4) + ATAN(XF(7)/PSUM12) + GAUSS(0.,STD(4))
         H(1) = HH(12)
         H(4) = HH(13)
         H(7) = HH(14)
       ENDIF
C
C     AZIMUTH RATE
      IF (IMEAS .EQ. 5) THEN
         ZRES = -(XF(1)*XF(5) - (XF(2)-XTRAJ(16))*XF(4))/PSUM
     1        +XTRAJ(5) + GAUSS(0.,STD(5))
         H(1) = HH(15)
         H(2) = HH(16)
         H(4) = HH(17)
         H(5) = HH(18)
      ENDIF
C
C     ELEVATION RATE
      IF(IMEAS .EQ. 6) THEN
         ZRES = (XF(8)*PSUM - XF(7)*(XF(1)*(XF(2)-XTRAJ(16))) +
     1          XF(4)*XF(5))/
     2          (SUMSQ*PSUM12) + GAUSS(0.,STD(6)) + XTRAJ(6)
         H(1) = HH(19)
         H(2) = HH(20)
         H(4) = HH(21)
         H(5) = HH(22)
         H(7) = HH(23)
         H(8) = HH(24)
      ENDIF
C
       RETURN
C
C     ******************
C     * HRZ ENTRY POINT *
C     ******************
C
      ENTRY  HRZO (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, NTR, PF,
     +             IMEAS, M, H, RF, ZRES)
      STD(1) = SQRT(RFIN(1))
      STD(2) = SQRT(RFIN(2))
      STD(3) = SQRT(RFIN(3))
      STD(4) = SQRT(RFIN(4))
      STD(5) = SQRT(RFIN(5))
      STD(6) = SQRT(RFIN(6))
      RETURN
      END
*DECK TRAJO
      SUBROUTINE  TRAJO (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ)
C
C
C +++ =TRAJO=, A USER-WRITTEN SUBROUTINE FOR SOFE.
C +++ CALLED FROM USER ROUTINES FOR INTERNAL TRAJECTORY
C +++ CONSTRUCTION.  MANDATORY FOR READING TAPE HEADER
C +++ AT TO IF TRAJECTORY IS STORED EXTERNALLY ON TAPE.
C
      DIMENSION  XF(NF), XS(NS), XTRAJ(NXTJ)
      CHARACTER TITLE*40,MODE*20,NAME*30
C
C     TRAJO READS AND ECHOS TITLE,NAME, AND MODE
      READ(3)TITLE
      READ(3)NAME
      READ(3)MODE
      IF(IRUN .NE. 1)RETURN
C     PRINT TITLE, NAME, AND MODE
      WRITE(6,1000)TITLE
      WRITE(6,1000)NAME
      WRITE(6,1000)MODE
      RETURN
 1000 FORMAT(A40)
      END
*DECK SNOYS
      SUBROUTINE  SNOYS (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ)
C
C
C +++ =SNOYS=, A USER-WRITTEN SUBROUTINE FOR SOFE.
C +++ CALLED AT DTNOYS INTERVALS.  PERTURBS TRUTH STATES WITH
C +++ RANDOM NOISE SAMPLES TO SIMULATE ACCUMULATED EFFECT OF
C +++ PROCESS DRIVING NOISE OVER THE INTERVAL DTNOYS.
C +++ FOR THIS FILTER, THE ABOVE DOES NOT PERTAIN.  RATHER,
C +++ THE TRUTH STATES ARE READ INTO XS AT DTNOYS INTERVALS.
C
      DIMENSION  XF(NF), XS(NS), XTRAJ(NXTJ)
C
C     FIRST OBTAIN TRUE STATES XS FROM XTRAJ
C
      XS(1)=XTRAJ(7)
      XS(2)=XTRAJ(8)
      XS(3)=XTRAJ(9)
      XS(4)=XTRAJ(10)
      XS(5)=XTRAJ(11)
      XS(6)=XTRAJ(12)
      XS(7)=XTRAJ(13)
      XS(8)=XTRAJ(14)
      XS(9)=XTRAJ(15)
C
      RETURN
C
C     *********************
C     * SNOYS ENTRY POINT *
C     *********************
C
      ENTRY  SNOYSO (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ)
C
      RETURN
      END
*DECK USRIN
      SUBROUTINE  USRIN
C
C
C +++ =USRIN=, A USER-WRITTEN SUBROUTINE FOR SOFE.
C +++ CALLED ONCE AT THE BEGINNING OF THE PROBLEM.  USEFUL
C +++ FOR READING IN AND ECHOING PARAMETERS THAT DEFINE
C +++ USER'S PROBLEM.  TRACKER PROBLEM WITH LINEAR DYNAMICS AND
C +++ NONLINEAR MEASUREMENTS.
C
      COMMON/QF/QFIN(3)
      COMMON/RFCOM/RFIN(6)
      COMMON/ROTCOM/ROTATE(9,9)
      COMMON/NCOM/NF,NS,M,NZF,NZQ,NXTJ,NTR,NALL
      NAMELIST/IN/QFIN,RFIN
      READ(5,IN)
      WRITE(6,IN)
C     ZEROIZE ROTATE MATRIX (USED IN SUBROUTINE SOFMOD1)
      DO 50 I=1,NF
         DO 40 J=1,NF
         ROTATE(I,J)=0.
   40    CONTINUE
   50 CONTINUE
      RETURN
      END
*DECK XFDOT
      SUBROUTINE  XFDOT (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, NTR, PF,
     +                           XDOT)
C
C
C +++ =XFDOT=, A USER-WRITTEN SUBROUTINE FOR SOFE.
C +++ CALLED ON DEMAND OF THE INTEGRATOR.  SUPPLIES THE
C +++ DERIVATIVE OF THE ESTIMATED FILTER STATE VECTOR (XFDOT=F(XF,T)).
C
      COMMON/OLD/TURNOLD,PHIOLD
      DIMENSION  XF(NF), XS(NS), XTRAJ(NXTJ), PF(NTR), XDOT(NF)
C
      XDOT(1) = XF(2) - XTRAJ(16)
      XDOT(2) = XF(3)
      XDOT(3) = -7.*XF(3)
      XDOT(4) = XF(5) - XTRAJ(17)
      XDOT(5) = XF(6)
      XDOT(6) = -7.*XF(6)
      XDOT(7) = XF(8) - XTRAJ(18)
      XDOT(8) = XF(9)
      XDOT(9) = -7.*XF(9)
      RETURN
C
C     *********************
C     * XFDOT ENTRY POINT *
C     *********************
C
      ENTRY  XFDOTO (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, NTR, PF,
     +                   XDOT)
C
C     SAVE INITIAL VALUES OF TURNOLD AND PHIOLD FOR SUBROUTINE SOFMOD1.
      TURNOLD = XTRAJ(19)
      PHIOLD = XTRAJ(20)
C     GET INITIAL ESTIMATES FROM EXTERNAL TRAJECTORY
      XF(1)=XTRAJ(7)
      XF(2)=XTRAJ(8)
      XF(3)=XTRAJ(9)
      XF(4)=XTRAJ(10)
      XF(5)=XTRAJ(11)
      XF(6)=XTRAJ(12)
      XF(7)=XTRAJ(13)
      XF(8)=XTRAJ(14)
      XF(9)=XTRAJ(15)
      RETURN
      END
*DECK XSDOT
      SUBROUTINE  XSDOT (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, XDOT)
C
C
C +++ =XSDOT=, A USER-WRITTEN SUBROUTINE FOR SOFE.
C +++ CALLED ON DEMAND OF THE INTEGRATOR.  SUPPLIES THE HOMOGENEOUS
C +++ PART OF THE DERIVATIVE OF THE TRUTH STATE VECTOR.  SINCE THE
C +++ TRUTH STATE XS IS OBTAINED FROM THE EXTERNAL TRAJECTORY,
C +++ JUST DEFAULT.
C
```

```
      DIMENSION  XF(NF), XS(NS), XTRAJ(NXTJ), XDOT(NS)
C
      XDOT(1) = 0.
      XDOT(2) = 0.
      XDOT(3) = 0.
      XDOT(4) = 0.
      XDOT(5) = 0.
      XDOT(6) = 0.
      XDOT(7) = 0.
      XDOT(8) = 0.
      XDOT(9) = 0.
      RETURN
C
C     *********************
C     * XSDOT ENTRY POINT *
C     *********************
C
      ENTRY  XSDOTO (IRUN, T, NF, NS, NXTJ, XF, XS, XTRAJ, XDOT)
C
      XS(1) = XTRAJ(7)
      XS(2) = XTRAJ(8)
      XS(3) = XTRAJ(9)
      XS(4) = XTRAJ(10)
      XS(5) = XTRAJ(11)
      XS(6) = XTRAJ(12)
      XS(7) = XTRAJ(13)
      XS(8) = XTRAJ(14)
      XS(9) = XTRAJ(15)
      RETURN
      END
```

SOFE Job Control

```
APQ120,P3.
USER,T850420,ROSSULD.
CHARGE,*.
SETTL,1000.
GET,USERBNA.
GET,SOFEB.
GET,TAPE5=SOFEDAT.
GET,TAPE3=OUTBEAM.
GET,MTHLIB7.
LIBRARY,MTHLIB7.
ATTACH,TAPE4=KFTEMP/M=W.
LOAD(USERBNA,SOFEB).
EXECUTE(,*PL=100000).
```

SOFEPL Job Control

```
APQPLOT,P3.
USER,T850420,ROSSULD.
CHARGE,*.
SETTL,750.
RETURN,META.
GET,LGO=SOFEPLB/UN=V780355.
ATTACH,DISSPLA/UN=APPLIB.
ATTACH,TAPE4=KFTEMP.
GET,TAPE5=SPLDATB.
ATTACH,META=META1/M=W.
LDSET,LIB=DISSPLA,PRESET=0.
LGO(,*PL=100000).
```

APPENDIX D

SOFE Modification

As explained in Section 4.3.3, the state estimate and filter covariance must be rotated into the update frame at time t_i^+ from the propagation frame at time t_i^- to compensate for any changes in fighter heading, pitch, or roll during the interval from t_{i-1}^+ to t_i^-. The rotations are accomplished according to the following relations:

$$\underline{\hat{x}}(t_i^-)_{new} = \underline{C}\underline{\hat{x}}(t_i^-) \qquad \text{(D-1)}$$

$$\underline{P}(t_i^-)_{new} = \underline{C}\underline{P}(t_i^-)\underline{C}^T \qquad \text{(D-2)}$$

where $\underline{C}$ is a direction cosine matrix. Using Equation (2-4), $\underline{C}$ is formed as

$$[\underline{C}] = [\Delta\phi][\Delta\theta][\Delta\Psi] \qquad \text{(D-3)}$$

For the truth model used, $\Delta\theta$ equals zero so the above relation reduces to

$$[\underline{C}] = [\Delta\phi][\Delta\Psi] \qquad \text{(D-4)}$$

where

$$[\Delta\Psi] = \begin{bmatrix} c(\Delta\Psi) & s(\Delta\Psi) & 0 \\ -s(\Delta\Psi) & c(\Delta\Psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$[\Delta\phi] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\Delta\phi) & s(\Delta\phi) \\ 0 & -s(\Delta\phi) & c(\Delta\phi) \end{bmatrix}$$

c = cosine, and

s = sine

For a three-by-three matrix

$$[\underline{C}] = \begin{bmatrix} c(\Delta\Psi) & s(\Delta\Psi) & 0 \\ -c(\Delta\phi)s(\Psi) & c(\Delta\phi)c(\Delta\Psi) & s(\Delta\phi) \\ s(\Delta\phi)s(\Delta\Psi) & -s(\Delta\phi)c(\Delta\Psi) & c(\Delta\phi) \end{bmatrix} \quad \text{(D-5)}$$

Similarly, expanding to a nine-by-nine matrix results in

$$[\underline{C}] = \begin{bmatrix} C11 & 0 & 0 & C12 & 0 & 0 & 0 & 0 & 0 \\ 0 & C11 & 0 & 0 & C12 & 0 & 0 & 0 & 0 \\ 0 & 0 & C11 & 0 & 0 & C12 & 0 & 0 & 0 \\ C21 & 0 & 0 & C22 & 0 & 0 & C23 & 0 & 0 \\ 0 & C21 & 0 & 0 & C22 & 0 & 0 & C23 & 0 \\ 0 & 0 & C21 & 0 & 0 & C22 & 0 & 0 & C23 \\ C31 & 0 & 0 & C32 & 0 & 0 & C33 & 0 & 0 \\ 0 & C31 & 0 & 0 & C32 & 0 & 0 & C33 & 0 \\ 0 & 0 & C31 & 0 & 0 & C32 & 0 & 0 & C33 \end{bmatrix} \quad \text{(D-6)}$$

where

$C11 = c(\Delta\Psi)$

$C12 = s(\Delta\Psi)$

$C21 = -c(\Delta\phi)s(\Psi)$

$C22 = c(\Delta\phi)c(\Delta\Psi)$

$C23 = s(\Delta\phi)$

$C31 = s(\Delta\phi)s(\Delta\Psi)$

$C32 = -s(\Delta\phi)c(\Delta\Psi)$

$C33 = c(\Delta\phi)$

Finally, page D-4 illustrates where the rotation routine is inserted in SOFE and pages D-5 through D-8 illustrate how Equation (D-6) is incorporated into the source code.

SOFE Modification Call

```
      END
*DECK MCRUN
      SUBROUTINE  MCRUN (T)
C
C +++ =MCRUN= IS THE CONTROLLING ROUTINE FOR A MONTE CARLO RUN. =MCRUN=
C +++ IS EITHER INTEGRATING OR UPDATING.  WHEN INTEGRATING, =MCRUN=
C +++ RELIES ON SUBROUTINE =ADVANS= TO SCHEDULE AND ACCOMPLISH ALL (SIX)
C +++ SOFE EVENTS EXCEPT MEASUREMENT UPDATE.  AT AN UPDATE TIME, T-,
C +++ =ADVANS= RETURNS TO =MCRUN= WHERE THE VARIOUS UPDATE SUBROUTINES
C +++ ARE CALLED IN SEQUENCE TO REACH TIME T+C.  WHEN T REACHES TF, THE
C +++ FINAL TIME OF A RUN, =MCRUN= RETURNS CONTROL TO =SOFE=.
C
      COMMON  A(1)
      COMMON / ICOM / ICONT, ISEED, IPASS, IPRRUN, IPGSIZ, IRUN, IDATA,
     +                SEED
      COMMON / IPOINT / IY, IXS, IXF, IPF, ISF, IFIND, IQIND, IFF, IQQ,
     +                  IXSO, IXFO, IPFO, IYDOT, IXSDOT, IXFDOT, IPFDOT,
     +                  IDEWK, IH, ISTHT, IG, ITKNOT, IUKNOT, IUXTJ,
     +                  ICOEF
      COMMON / NCOM / NF, NS, M, NZF, NZQ, NXTJ, NTR, NALL
      COMMON / OTHER / ZRES, LXTJ, IMEAS, ALPHA
      LOGICAL  ENDFLG, MSFLG
C
C *** PROPAGATE XS, XF AND PF BY NUMERICAL INTEGRATION UNTIL
C *** T- AND/OR TF ARE REACHED.
C
  100 CONTINUE
      CALL  ADVANS (T, MSFLG, ENDFLG)
      IF  (MSFLG)  THEN
****************************SOFE MODIFICATION*************************
*****************************SOFE MODIFICATION************************
      CALL SOFMOD1(NF,NXTJ,A(IXF),A(IDATA),A(IPF),NTR)
****************************THAT'S IT*********************************
C
C *** TIME IS T- .  UPDATE XF- AND PF- .  BEGIN BY FINDING SQUARE ROOT
C *** SF- OF PF- .  THEN SEQUENCE THRU M MEASUREMENTS, ONE AT A TIME.
C
          CALL  OUT (T, 6)
          CALL  PSQRT (NF, NTR, A(IPF), IERFLG, A(ISF))
          IF  (IERFLG .EQ. 2)  CALL ERROUT (11, 'MCRUN')
          DO  110 II = 1, M
              IMEAS = II
C
C *** GET MEASUREMENT DATA FROM USER-ROUTINE =HRZ=.
C *** UPDATE SF- AND XF- IN XSPLUS.  SQUARE SF+ SO PF IS CURRENT.
C
              CALL  HRZ (IRUN, T, NF, NS, NXTJ, A(IXF), A(IXS),
     +                   A(IDATA), NTR, A(IPF), IMEAS, M,
     +                   A(IH), RMEAS, ZRES)
              IF  (RMEAS .GE. 0.)  THEN
                  CALL  XSPLUS (NF, NTR, A(IH), RMEAS, ZRES,
     +                          A(1), A(ISTHT), A(IG), ALPHA,
     +                          A(IXF), A(ISF))
```

SOFE Modification

```
*DECK SOFMOD1
      SUBROUTINE SOFMOD1(NF,NXTJ,AIXF,AIDATA,AIPF,NTR)
C
C +++ =SOFMOD1= REORIENTATES THE REFERENCE FRAME AT TIME T(I-1) TO
C +++ THAT AT TIME T(I).  THIS IS ACCOMPLISHED BY DIRECTION COSINE
C +++ MATRICES.  FIRST, HEADING AND ROLL ANGLES (EXTERNAL TRAJECTORY
C +++ VALUES) AT TIME T(I) ARE DIFFERENCED WITH THOSE AT TIME T(I-1)
C +++ TO FORM DELTA ANGLES TO USE IN THE DIRECTION COSINE MATRICES.
C +++ THE DIRECTION COSINE MATRIX, ROTATE, IS THEN APPLIED TO THE
C +++ FILTER'S XF- AND PF-.
C
C     WRITTEN BY CAPT. ROSS ANDERSON
C     SUBROUTINE CALLS CPYVM, CPYMV, MMUL, MTRN WRITTEN BY OTHERS
C
      COMMON/OLD/TURNOLD,PHIOLD
      COMMON/ROTCOM/ROTATE(9,9)
C     COMMON/MOD1/PMAT,WRKSPCX,WRKSPCP,AIXFM,ROTTRAN
      DIMENSION AIXF(9),AIDATA(20),AIPF(45),PMAT(9,9),
     1 WRKSPCX(9,1),WRKSPCP(9,9),AIXFM(9,1),
     2 ROTTRAN(9,9)
C
C *** DETERMINE DELTA ROTATION ANGLES (INITIAL VALUES FROM XFDOTO)
      DELTURN = AIDATA(19) - TURNOLD
      DELPHI  = AIDATA(20) - PHIOLD
      TURNOLD = AIDATA(19)
      PHIOLD  = AIDATA(20)
C
C *** FORM DIRECTION COSINE MATRIX, ROTATE(SET=0. IN SUBROUTINE USRIN)
      ROTATE(1,1) = COS(DELTURN)
      ROTATE(2,2) = ROTATE(1,1)
      ROTATE(3,3) = ROTATE(1,1)
      ROTATE(1,4) = SIN(DELTURN)
      ROTATE(2,5) = ROTATE (1,4)
      ROTATE(3,6) = ROTATE (1,4)
      ROTATE(1,7) = 0.
      ROTATE(2,8) = ROTATE(1,7)
      ROTATE(3,9) = ROTATE(1,7)
      ROTATE(4,1) = -COS(DELPHI)*SIN(DELTURN)
      ROTATE(5,2) = ROTATE(4,1)
      ROTATE(6,3) = ROTATE(4,1)
      ROTATE(4,4) = COS(DELPHI)*COS(DELTURN)
      ROTATE(5,5) = ROTATE(4,4)
      ROTATE(6,6) = ROTATE(4,4)
      ROTATE(4,7) = SIN(DELPHI)
      ROTATE(5,8) = ROTATE(4,7)
      ROTATE(6,9) = ROTATE(4,7)
      ROTATE(7,1) = SIN(DELPHI)*SIN(DELTURN)
      ROTATE(8,2) = ROTATE(7,1)
      ROTATE(9,3) = ROTATE(7,1)
      ROTATE(7,4) = -SIN(DELPHI)*COS(DELTURN)
      ROTATE(8,5) = ROTATE(7,4)
      ROTATE(9,6) = ROTATE(7,4)
      ROTATE(7,7) = COS(DELPHI)
      ROTATE(8,8) = ROTATE(7,7)
      ROTATE(9,9) = ROTATE(7,7)
C
C *** NOW ROTATE XF (XF = ROTATE*XF)
C
C     FIRST, CONVERT XF INTO A COLUMN VECTOR
C
      DO 50 I=1,NF
         AIXFM(I,1)=AIXF(I)
  50  CONTINUE
C
C     SECOND, MULTIPLY
      CALL MMUL(ROTATE,AIXFM,AIXFM,
     1 WRKSPCX,NF,NF,1)
C
C     FINALLY, CONVERT XF BACK INTO A ROW VECTOR
      DO 60 I=1,NF
         AIXF(I)=AIXFM(I,1)
  60  CONTINUE
C
C *** NOW ROTATE PF (PF = ROTATE*PF*(ROTATE TRANSPOSED))
C
C     FIRST, PF MUST BE CONVERTED FROM A VECTOR TO A MATRIX
      CALL CPYVM(NF,NTR,AIPF,PMAT)
C
C     SECOND, MULTIPLY, PMAT = ROTATE*PMAT
      CALL MMUL(ROTATE,PMAT,PMAT,
     1 WRKSPCP,NF,NF,NF)
C
C     THIRD, MULTIPLY, PMAT = PMAT*ROTTRAN (ROTTAN = ROTATION TRANSPOSED
      CALL MTRN(ROTATE,ROTTRAN,NF,NF)
      CALL MMUL(PMAT,ROTTRAN,PMAT,
     1 WRKSPCP,NF,NF,NF)
C
C     FINALLY, PMAT MUST BE CONVERTED FROM A MATRIX BACK TO A VECTOR
      CALL CPYMV (NF,NTR,PMAT,AIPF)
      RETURN
      END
*DECK CPYMV
      SUBROUTINE  CPYMV (N, L, A, R)
C
C     ...........................................................
C
C        SUBROUTINE CPYMV
C
C        PURPOSE
C           COPY THE UPPER TRIANGULAR ELEMENTS OF A SQUARE MATRIX INTO
C           A VECTOR.  TO ILLUSTRATE, THE 4X4 MATRIX A IS COPIED TO
C           THE 10X1 VECTOR R AS FOLLOWS:
C                  ABCD
C              A = EFGH  ===>>  R=(A,B,F,C,G,K,D,H,L,P)
C                  IJKL
C                  MNOP
```

```
C
C        USAGE
C           CALL CPYMV(N,L,A,R)
C
C        DESCRIPTION OF PARAMETERS
C           N - INPUT, NUMBER OF ROWS (COLUMNS) IN SQUARE MATRIX A
C           L - INPUT, NUMBER OF ELEMENTS IN VECTOR R (=N(N+1)/2)
C           A - INPUT, NAME OF INPUT MATRIX
C           R - OUTPUT, NAME OF OUTPUT VECTOR
C
C        REMARKS
C           L MUST EQUAL N(N+1)/2, THE NUMBER OF ELEMENTS IN THE
C           UPPER TRIANGULAR PORTION OF A.
C
C        SUBPROGRAMS REQUIRED
C           NONE
C
C     ...................................................................
C
      DIMENSION  A(N, N), R(L)
C
      K = 0
      DO  110 J = 1, N
          DO  100 I = 1, J
              K = K + 1
              R(K) = A(I, J)
  100     CONTINUE
  110 CONTINUE
      RETURN
      END
*DECK CPYVM
      SUBROUTINE  CPYVM (N, L, R, A)
C
C     ...................................................................
C
C        SUBROUTINE CPYVM
C
C        PURPOSE
C           COPY THE ELEMENTS OF A VECTOR TO FORM A SYMMETRIC MATRIX.
C           TO ILLUSTRATE, THE 10X1 VECTOR R IS COPIED INTO THE
C           4X4 MATRIX A AS FOLLOWS:
C                                                     ABDG
C              R = (A,B,C,D,E,F,G,H,I,J)  ===>>  A = BCEH
C                                                     DEFI
C                                                     GHIJ
C
C        USAGE
C           CALL CPYVM(N,L,R,A)
C
C        DESCRIPTION OF PARAMETERS
C           N - INPUT, NUMBER OF ROWS (COLUMNS) IN SQUARE MATRIX A
C           L - INPUT, NUMBER OF ELEMENTS IN VECTOR R (=N(N+1)/2)
C           R - INPUT, NAME OF INPUT VECTOR
```

```
C           A - OUTPUT, NAME OF OUTPUT MATRIX
C
C        REMARKS
C           L MUST EQUAL N(N+1)/2, THE NUMBER OF ELEMENTS IN THE
C           UPPER TRIANGULAR PORTION OF A.
C
C        SUBPROGRAMS REQUIRED
C           NONE
C
C     ..............................................................
C
      DIMENSION  A(N, N), R(L)
C
      K = 0
      DO  110 J = 1, N
          DO  100 I = 1, J
              K = K + 1
              A(J, I) = R(K)
              A(I, J) = R(K)
  100     CONTINUE
  110 CONTINUE
      RETURN
      END
```

Appendix E

Equivalent Discrete-Time Algorithm

```
      PROGRAM EXEC (SAS Program)
C +++ OBJECTIVE -
C        -A COMPUTER AIDED DESIGN PACKAGE TO AID IN THE
C         DEVELOPMENT, DESIGN, TUNING, AND EVALUATION OF AN EXTENDED
C         KALMAN FILTER(EKF) FOR AN AIR-TO-AIR FIRE CONTROL SYSTEM.
C         THE FILTER IS BASED ON A LINEAR DYNAMICS MODEL AND NON-
C         LINEAR MEASUREMENTS.  A CONSTANT STATE TRANSITION MATRIX
C         (STM) IS EMPLOYED TO PROVIDE FILTER PROPAGATION.  OVER-
C         ALL, THE FILTER PROVIDES BODY AXIS REFERENCED ESTIMATES OF
C         CARTESIAN VARIABLES OF POSITION, VELOCITY, AND ACCELERATION.
C        -AN EQUIVALENT DISCRETE TIME DESIGN OF A SPECIFIC EKF
C        -TO BE USED TO REPLACE SOFE AND SOFEPL FOR A SPECIFIC
C         APPLICATION.
C        -EKF BASED ON A NINE STATE REDUCED ORDER MODEL OF
C         POSITION(3), VELOCITY(3), AND ACCELERATION(3).
C        -DESIGNED SO APPLICABLE PORTIONS OF SOFTWARE CAN BE
C         'LIFTED OUT' FOR EVENTUAL IMPLEMENTATION ON THE
C         F-4 E/G MODEL AIRCRAFT.
C
C +++ AUTHOR - CAPT ROSS ANDERSON
C            - *DECK GAUSS FROM SOFE(SLIGHTLY MODIFIED)
C
C +++ EXTERNAL PROGRAMS REQUIRED
C         1.  HILL5 - GENERATES EXTERNAL TRAJECTORY, TRUTH STATES,
C                     AND MEASUREMENTS.  HILL5 IS
C                     RUN PREVIOUS TO THIS SIMULATION AND THE
C                     RESULTS STORED FOR USE BY THIS PROGRAM.  (DATA
C                     IS STORED IN AN UNFORMATTED FORMAT)
C
C         2. MATHLIB - MATRIX ROUTINES PACKAGE WRITTEN
C                     BY AFWAL/AAAN
C
C         3. POSTPROCESSOR PLOTTING PACKAGE - SUBROUTINE POSTPRC
C                     GENERATES THE AVERAGE OF THE DIFFERENCE E OF
C                     THE TRUTH STATE AND THE FILTER STATE, THE
C                     STANDARD DEVIATION SE OF THIS DIFFERENCE, AND
C                     THE PLUS AND MINUS SQUARE ROOT VALUE OF THE
C                     DIAGONAL OF THE FILTER COMPUTED COVARIANCE.
C                     A POSTPROCESSOR PLOTTING PACKAGE IS REQUIRED
C                     TO GRAPHICALLY PLOT THE RESULTS.  BASICALLY,
C                     ANY PLOTTING PACKAGE CAN BE USED.  FOR PLOTS
C                     INCLUDED IN THESIS WORK, A PLOT M FUNCTION
C                     FROM A PROC FILE CALLED 'PROCFIL' IS USED.
C                     TO GET PROCFIL ON AN AIR FORCE INSTITUTE OF
C                     TECHNOLOGY(AFIT) CYBER ACCOUNT, THE FOLLOWING
C                     STEPS MUST BE TAKEN:
C                       A. LOGIN TO ACCOUNT
C                       B. TYPE IN AFTER THE PROMPT AND THEN 'RETURN'
C                          GET,PROCFIL/UN=T840115
C                          REPLACE,PROCFIL
C                          UPROC,PROCFIL
C                          BEGIN,PV,,XXX   (WHERE XXX IS YOUR USER
C                                                       JOB NUMBER)
```

```
C
C +++ EXTERNAL FUNCTIONS REQUIRED
C
C     1. STANDARD MATH ROUTINES: SQRT, /, *, ETC.
C
C +++ DEFINITION OF TERMS/VARIABLES
C
       COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1           IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
C     ******************************
C     ****  INITIALIZE PROBLEM  ****
C     ******************************
C
      CALL INITPRB
C
C     **************************
C     ****  INITIALIZE RUN  ****
C     **************************
C
  100 CALL INITRUN
C
C     ***********************************************
C     ****  READ EXTERNAL DATA FOR CURRENT TIME  ****
C     ***********************************************
C
  200 CALL READEXT
C
C     *************************************************
C     ****  GENERATE NOISE TO ADD TO MEASUREMENTS  ****
C     *************************************************
C
C     DONE THROUGH REAL FUNCTION GAUSS
C
C     *************************************
C     ****  KALMAN FILTER PROPAGATION  ****
C     *************************************
C
      CALL PROPAG
C
C     ********************************
C     ****  KALMAN FILTER UPDATE  ****
C     ********************************
C
      CALL UPDATE
C
C     ***************************
C     ****  SAVE STATISTICS  ****
C     ***************************
C
      CALL SAVSTAT
C
************************************************
****  ANOTHER TIME PROPAGATION AND UPDATE?  ****
```

```
*************************************************
C
      IF (T .LT. TF) GO TO 200
C
C     ************************
C     ****  RUN COMPLETE  ****
C     ************************
C
      IRUN = IRUN + 1
      IF (IRUN .LE. IPASS)  GO TO 100
C
C     **********************************************
C     ****  PROBLEM COMPLETE, POST PROCESS DATA  ****
C     **********************************************
C
      CALL POSTPRC
C
      STOP   'PROBLEM DONE'
      END
* DECK INITPRB
      SUBROUTINE INITPRB
C
C +++ =INITPRB= IS CALLED ONCE AT THE BEGINNING OF EACH PROBLEM.
C +++ EXECUTION TO INITIALIZE THE PROBLEM.  INITPRB SETS PROBLEM
C +++ VARIABLES, READS PROBLEM VARIABLES, ECHOS PROBLEM VARIABLES,
C +++ CALCULATES A CONSTANT STM PHI, CALCULATES
C +++ PROBLEM STANDARD DEVIATION, SETS THE H MATRIX TO ZERO.
C *** SETS THE ROTATE MATRIX TO ZERO,  INITIALIZES
C *** SAVSTAT VARIABLES, AND CALCULATES A CONSTANT QD MATRIX.
C
      COMMON/QFCOM/QFIN(3)
      COMMON/QDCOM/QD(9,9)
      COMMON/TAUCOM/TAU(3)
      COMMON/RFCOM/RMAT(6,6)
      COMMON/ROTCOM/ROTATE(9,9)
      COMMON/SDEVCOM/STD(6)
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1              IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
      COMMON/SAVSTAT/XE(9),SUMXE(500,9,2),SUMPPL(500,9)
      COMMON/PHICOM/PHI(9,9)
      COMMON/HCOM/H(6,9)
      COMMON/INPCTRL/NTSIN
      DIMENSION RFIN(6)
      REAL H
      EQUIVALENCE(OUTDAT,OUTBEAM,OUTTAIL)
C
C *** SETUP INPUT AND OUTPUT TAPES
C
C     TAPE3     INPUT     UNFORMATTED   -EXTERNAL TRAJECTORY DATA
C     TAPE4     OUTPUT    UNFORMATTED   -OUTPUT THAT CAN BE
C                                        POSTPROCESSED
C     TAPE5     INPUT     FORMATTED     -TITLES, R, Q, TAU, AND
C                                        INITIALIZATION PARAMETERS
```

```
C                                        IPASS AND IBUG
C     TAPE6     OUTPUT    FORMATTED     -ECHO PROGRAM AND IBUG DATA
C
      OPEN(UNIT=3,FILE='OUTDAT',STATUS='OLD',FORM='UNFORMATTED')
      OPEN(UNIT=4,FILE='PLTDATA',STATUS='NEW',FORM='UNFORMATTED')
      OPEN(UNIT=5,FILE='SADATA',STATUS='OLD')
      OPEN(UNIT=6,FILE='ECHO',STATUS='NEW')
C
C *** SET PROBLEM VALUES
C
      NF=9
      NS=9
      M=6
      NXTJ=20
      TO=0.
      TF=12.0
      DTMEAS=.1
      IPASS=20
      ISEED=77
      IRUN=1
      IBUG1=1
      IBUG2=1
      IBUG3=1
      IBUG4=1
      IBUG5=1
      IBUG6=1
      IBUG7=1
      NTSIN=5
C
C *** NOTE THE ABOVE IBUG PARAMETERS CONTROL OUTPUT
C     TO TAPE6.  NTSIN CONTROLS THE NUMBER OF TIME
C     STEPS READ FROM THE EXTERNAL TRAJECTORY TAPE.
C     EXAMPLE, IF NTSIN=5, THE FIRST EXTERNAL DATA
C     RECORD IS READ THEN EVERY FIFTH ONE THEREAFTER.
C     NTS MUST BE COORDINATED WITH DTMEAS OF THIS
C     PROGRAM AND DT OF PROGRAM HILL5
C
C *** READ NAMELIST VARIABLE PROBLEM VALUES
C
      NAMELIST/PRBINIT/QFIN,RFIN,TAU,IPASS,IBUG1,
     1        IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7,
     2        NTSIN,TF
      READ(5,PRBINIT)
C
C *** RMAT(M,M) TO ZERO
C
      DO 30 I=1,M
          DO 20 J=1,M
          RMAT(I,J)=0.
   20     CONTINUE
   30 CONTINUE
C
C *** FILL IN NONZERO RMAT VALUES
```

```
C
      RMAT(1,1)=RFIN(1)
      RMAT(2,2)=RFIN(2)
      RMAT(3,3)=RFIN(3)
      RMAT(4,4)=RFIN(4)
      RMAT(5,5)=RFIN(5)
      RMAT(6,6)=RFIN(6)
C
C
C *** ECHO PROBLEM VALUES
      WRITE(6,*)'*****THE FOLLOWING ARE SET IN SUB INITPRB AS:*****'
      WRITE(6,*)
C
      WRITE(6,*)'NF=',NF
      WRITE(6,*)'NS=',NS
      WRITE(6,*)'M=',M
      WRITE(6,*)'NXTJ=',NXTJ
      WRITE(6,*)'TO=',TO
      WRITE(6,*)'TF=',TF
      WRITE(6,*)'DTMEAS=',DTMEAS
      WRITE(6,*)'ISEED=',ISEED
      WRITE(6,*)'IRUN=',IRUN
      WRITE(6,*)'IPASS=',IPASS
      WRITE(6,*)'Q(1)=',QFIN(1)
      WRITE(6,*)'Q(2)=',QFIN(2)
      WRITE(6,*)'Q(3)=',QFIN(3)
      WRITE(6,*)'RFIN(1)=',RFIN(1)
      WRITE(6,*)'RFIN(2)=',RFIN(2)
      WRITE(6,*)'RFIN(3)=',RFIN(3)
      WRITE(6,*)'RFIN(4)=',RFIN(4)
      WRITE(6,*)'RFIN(5)=',RFIN(5)
      WRITE(6,*)'RFIN(6)=',RFIN(6)
      WRITE(6,*)'TAU(1)=',TAU(1)
      WRITE(6,*)'TAU(2)=',TAU(2)
      WRITE(6,*)'TAU(3)=',TAU(3)
      WRITE(6,*)'IBUG1=',IBUG1
      WRITE(6,*)'IBUG2=',IBUG2
      WRITE(6,*)'IBUG3=',IBUG3
      WRITE(6,*)'IBUG4=',IBUG4
      WRITE(6,*)'IBUG5=',IBUG5
      WRITE(6,*)'IBUG6=',IBUG6
      WRITE(6,*)'IBUG7=',IBUG7
      WRITE(6,*)
C
C *** CALCULATE CONSTANT STATE TRANSITION MATRIX(STM).  IN
C *** REALITY, THIS MATRIX CAN BE PRECOMPUTED FOR A CONSTANT
C *** UPDATE RATE (DTMEAS).  THEN, ONLY THE PRECOMPUTED VALUES
C *** HAVE TO BE STORED.  CALCULATIONS ARE INCLUDED HERE FOR
C *** FLEXIBILITY, IE, IT IS VERY EASY TO CHANGE TAU AND
C *** DTMEAS.
C
C     SET ALL VALUES OF STM PHI TO ZERO
C
```

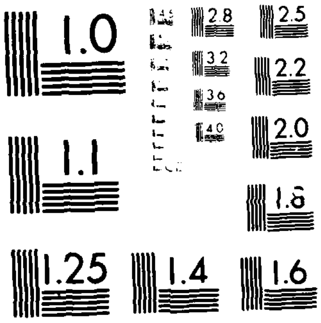
```
      DO 50 I=1,NF
         DO 40 J=1,NF
         PHI(I,J)=0.
   40    CONTINUE
   50 CONTINUE
C
C     CALCULATE NON-ZERO VALUES OF PHI.
C
      PHI(1,1)=1.
      PHI(1,2)=DTMEAS
      PHI(1,3)=TAU(1)**2*((1./TAU(1))*DTMEAS-1.+EXP(-((1./TAU(1))*
     1       DTMEAS)))
      PHI(2,2)=1.
      PHI(2,3)=TAU(1)*(1.-EXP(-((1./TAU(1))*DTMEAS)))
      PHI(3,3)=EXP(-((1./TAU(1))*DTMEAS))
      PHI(4,4)=1.
      PHI(4,5)=DTMEAS
      PHI(4,6)=TAU(2)**2*((1./TAU(2))*DTMEAS-1.+EXP(-((1./TAU(2))*
     1       DTMEAS)))
      PHI(5,5)=1.
      PHI(5,6)=TAU(2)*(1.-EXP(-((1./TAU(2))*DTMEAS)))
      PHI(6,6)=EXP(-((1./TAU(2))*DTMEAS))
      PHI(7,7)=1.
      PHI(7,8)=DTMEAS
      PHI(7,9)=TAU(3)**2*((1./TAU(3))*DTMEAS-1.+EXP(-((1./TAU(3))*
     1       DTMEAS)))
      PHI(8,8)=1.
      PHI(8,9)=TAU(3)*(1.-EXP(-((1./TAU(3))*DTMEAS)))
      PHI(9,9)=EXP(-((1./TAU(3))*DTMEAS))
C
C *** CALCULATE STANDARD DEVIATION
C
      STD(1) = SQRT(RFIN(1))
      STD(2) = SQRT(RFIN(2))
      STD(3) = SQRT(RFIN(3))
      STD(4) = SQRT(RFIN(4))
      STD(5) = SQRT(RFIN(5))
      STD(6) = SQRT(RFIN(6))
C
C *** SET ALL VALUES OF THE H, QD, AND ROTATE MATRICES TO ZERO
C
      DO 60 I=1,NF
         DO 55 J=1,NF
          IF (I .LE. M) THEN
          H(I,J)=0.
          ENDIF
         ROTATE(I,J)=0.
         QD(I,J)=0.
   55    CONTINUE
   60 CONTINUE
C
C *** SET INITIAL VALUES OF POSTPRC TO ZERO
C
```

```
      DO 65 I=1,500
          DO 64 J=1,NF
             SUMXE(I,J,1)=0.
             SUMXE(I,J,2)=0.
             SUMFPL(I,J) =0.
   64     CONTINUE
   65 CONTINUE
C
C *** CALCULATE QD.
C *** THE EXACT VALUE OF QD IS CALCULATED.  IN REALITY, QD
C *** CAN BE PRECOMPUTED AND VALUES STORED FOR IMPLEMENTATION
C *** ON A SYSTEM.  CALCULATIONS ARE INCLUDED HERE FOR
C *** FLEXIBILITY, IE, IT IS VERY EASY TO CHANGE VALUES OF
C *** TAU, QFIN, AND DTMEAS.
C
      QD(1,1)=((QFIN(1)*TAU(1)**5)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(1)))+
     1       2.*DTMEAS/TAU(1)+2.*DTMEAS**3/(3.*TAU(1)**3)-
     2       2.*DTMEAS**2/TAU(1)**2-(4.*DTMEAS/TAU(1))*
     3       EXP(-(DTMEAS/TAU(1))))
      QD(1,2)=((QFIN(1)*TAU(1)**4)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(1)))-
     1       2.*EXP(-(DTMEAS/TAU(1)))+(2.*DTMEAS/TAU(1))*
     2       EXP(-(DTMEAS/TAU(1)))-2.*DTMEAS/TAU(1)+
     3       DTMEAS**2/TAU(1)**2)
      QD(1,3)=((QFIN(1)*TAU(1)**3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(1)))-
     1       (2.*DTMEAS/TAU(1))*EXP(-(DTMEAS/TAU(1))))
      QD(2,2)=((QFIN(1)*TAU(1)**3)/2.)*(4.*EXP(-(DTMEAS/TAU(1)))-3.-
     1       EXP(-(2.*DTMEAS/TAU(1)))+2.*DTMEAS/TAU(1))
      QD(2,3)=((QFIN(1)*TAU(1)**2)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(1)))-
     1       2.*EXP(-(DTMEAS/TAU(1))))
      QD(3,3)=(QFIN(1)*TAU(1)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(1))))
      QD(2,1)=QD(1,2)
      QD(3,1)=QD(1,3)
      QD(3,2)=QD(2,3)
      QD(4,4)=((QFIN(2)*TAU(2)**5)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(2)))+
     1       2.*DTMEAS/TAU(2)+2.*DTMEAS**3/(3.*TAU(2)**3)-
     2       2.*DTMEAS**2/TAU(2)**2-(4.*DTMEAS/TAU(2))*
     3       EXP(-(DTMEAS/TAU(2))))
      QD(4,5)=((QFIN(2)*TAU(2)**4)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(2)))-
     1       2.*EXP(-(DTMEAS/TAU(2)))+(2.*DTMEAS/TAU(2))*
     2       EXP(-(DTMEAS/TAU(2)))-2.*DTMEAS/TAU(2)+
     3       DTMEAS**2/TAU(2)**2)
      QD(4,6)=((QFIN(2)*TAU(2)**3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(2)))-
     1       (2.*DTMEAS/TAU(2))*EXP(-(DTMEAS/TAU(2))))
      QD(5,5)=((QFIN(2)*TAU(2)**3)/2.)*(4.*EXP(-(DTMEAS/TAU(2)))-3.-
     1       EXP(-(2.*DTMEAS/TAU(2)))+2.*DTMEAS/TAU(2))
      QD(5,6)=((QFIN(2)*TAU(2)**2)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(2)))-
     1       2.*EXP(-(DTMEAS/TAU(2))))
      QD(6,6)=(QFIN(2)*TAU(2)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(2))))
      QD(5,4)=QD(4,5)
      QD(6,4)=QD(4,6)
      QD(6,5)=QD(5,6)
      QD(7,7)=((QFIN(3)*TAU(3)**5)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(3)))+
     1       2.*DTMEAS/TAU(3)+2.*DTMEAS**3/(3.*TAU(3)**3)-
     2       2.*DTMEAS**2/TAU(3)**2-(4.*DTMEAS/TAU(3))*
     3       EXP(-(DTMEAS/TAU(3))))
      QD(7,8)=((QFIN(3)*TAU(3)**4)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(3)))-
     1       2.*EXP(-(DTMEAS/TAU(3)))+(2.*DTMEAS/TAU(3))*
     2       EXP(-(DTMEAS/TAU(3)))-2.*DTMEAS/TAU(3)+
     3       DTMEAS**2/TAU(3)**2)
      QD(7,9)=((QFIN(3)*TAU(3)**3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(3)))-
     1       (2.*DTMEAS/TAU(3))*EXP(-(DTMEAS/TAU(3))))
      QD(8,8)=((QFIN(3)*TAU(3)**3)/2.)*(4.*EXP(-(DTMEAS/TAU(3)))-3.-
     1       EXP(-(2.*DTMEAS/TAU(3)))+2.*DTMEAS/TAU(3))
      QD(8,9)=((QFIN(3)*TAU(3)**2)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(3)))-
     1       2.*EXP(-(DTMEAS/TAU(3))))
      QD(9,9)=(QFIN(3)*TAU(3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(3))))
      QD(8,7)=QD(7,8)
      QD(9,7)=QD(7,9)
      QD(9,8)=QD(8,9)
      WRITE(6,*)
      WRITE(6,*)'PHI MATRIX'
      WRITE(6,*)
      DO 70 K=1,NF
         WRITE(6,100)(K,I,PHI(K,I),I=1,NF)
   70 CONTINUE
      WRITE(6,*)
      WRITE(6,*)'QD MATRIX'
      WRITE(6,*)
      DO 75 K=1,NF
         WRITE(6,100)(K,I,QD(K,I),I=1,NF)
   75 CONTINUE
      WRITE(6,*)
C
      RETURN
  100 FORMAT(9(I1,I1,E11.5,1X))
      END
*DECK INITRUN
      SUBROUTINE INITRUN
C
C +++ =INITRUN= IS CALLED ONCE AT THE BEGINNING OF EACH RUN TO SET
C +++ UP THE RUN.  FIRST, INITIAL VALUES OF PPLUS ARE
C +++ SET.  THEN, THE HEADER
C +++ INFORMATION IS STRIPPED OFF THE XTRAJ TAPE (TAPE3).
C +++ THIS TAPE INFORMATION IS ECHOED TO TAPE 6 IF IRUN=1.
C +++ ICOUNT IS SET EQUAL TO 0 FOR USE IN SAVSTAT.
C
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1               IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
      COMMON/UPDATE/ZRES(6,1),XPPLUS(9,1),PPLUS(9,9)
      COMMON/INITRUN/ICOUNT
      CHARACTER TITLE*40,MODE*20,NAME*30
C
C *** INITIALIZE PO(PPLUS) FOR EACH PASS(IPASS)
C
C *** SET P VALUES EQUAL TO ZERO
C
```

```
      DO 80 I=1,NF
         DO 70 J=1,NF
         PPLUS(I,J)=0.
   70    CONTINUE
   80 CONTINUE
C
C     SET NON-ZERO P VALUES
C
      PPLUS(1,1)=1000.
      PPLUS(2,2)=4000.
      PPLUS(3,3)=9300.
      PPLUS(4,4)=1000.
      PPLUS(5,5)=4000.
      PPLUS(6,6)=9300.
      PPLUS(7,7)=1000.
      PPLUS(8,8)=4000.
      PPLUS(9,9)=9300.
C
C     READ EXTERNAL TRAJECTORY TO POSITION POINTER ON DATA.
C
      REWIND 3
      READ(3) TITLE
      READ(3) NAME
      READ(3) MODE
      IF (IRUN .EQ. 1) THEN
         WRITE(6,1000) TITLE
          WRITE(4) TITLE
         WRITE(6,1000) NAME
          WRITE(4) NAME
         WRITE(6,1000) MODE
          WRITE(4) MODE
      ENDIF
      ICOUNT=0
      RETURN
 1000 FORMAT(A40)
      END
*DECK READEXT
      SUBROUTINE READEXT
C
C +++ =READEXT= IS CALLED FOR EVERY RUN FOR EVERY TIME VALUE
C +++ STORED ON THE EXTERNAL TRAJECTORY OUTPUT OF HILLS
C +++ UP TO TIME TF.
C +++ FILE OUTDAT IS READ FOR XTRAJ STORED DATA.
C +++ THE EXTERNAL TAPE TIME MUST EQUAL OR EXCEED TF.
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1                IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
      COMMON/READEXT/XTRAJ(20)
      COMMON/INPCTRL/NTSIN
      COMMON/JUNK/XJUNK(20)
C
C *** READ TIME AND NXTJ EXTERNAL DATA VALUES
      READ(3) T,(XTRAJ(I),I=1,NXTJ)
      INCNTRL=NTSIN-1
      DO 90 J=1,INCNTRL
         READ(3) TJUNK,(XJUNK(I),I=1,NXTJ)
   90 CONTINUE
      RETURN
      END
C
*DECK GAUSS
      REAL FUNCTION  GAUSS (MEAN, STD)
C
C     ..................................................................
C
C     REAL FUNCTION GAUSS
C
C     PURPOSE
C        SAMPLE A GAUSSIAN (NORMAL) DISTRIBUTION WHOSE MEAN AND
C        STANDARD DEVIATION ARE GIVEN
C
C     USAGE
C        SAMPLE = GAUSS(MEAN,STD)
C
C     DESCRIPTION OF PARAMETERS
C        MEAN  - DESIRED MEAN OF THE GAUSSIAN DISTRIBUTION
C        STD   - DESIRED STANDARD DEVIATION OF THAT DISTRIBUTION
C        GAUSS - THE VALUE OF THE COMPUTED GAUSSIAN RANDOM VARIABLE
C
C     SUBPROGRAMS REQUIRED
C        PORTABLE RANDOM NUMBER GENERATOR SELECTED:
C           NONE
C        NON-PORTABLE RANDOM NUMBER GENERATOR SELECTED:
C           RANF(D)
C              UNIFORM RANDOM NUMBER GENERATOR FOR CDC EQUIPMENT.
C              PRODUCES SAMPLES IN THE INTERVAL (0,1).
C           RANSET(N)
C              THE SEED INITIALIZATION ROUTINE ON THE CDC.
C
C     REMARKS
C        TWO RANDOM NUMBER GENERATORS ARE PROVIDED, ONE COMPLETELY
C        PORTABLE, THE OTHER NON-PORTABLE (CDC SPECIFIC).
C        THE PORTABLE GENERATOR PRODUCES THE SAME RANDOM NUMBER
C        SEQUENCE (TO 32 BITS) ON ALL MACHINES.
C        THE USER CHOOSES BETWEEN THEM BY HIS INITIAL ASSIGNMENT
C        OF ISEED, THE SEED USED TO GENERATE UNIFORM DEVIATES.
C        ISEED IS INPUT TO GAUSS THROUGH COMMON/INITPRB/.
C        IF ISEED IS INITIALLY POSITIVE, A PORTABLE GENERATOR IS
C        SELECTED.
C        IF ISEED IS NEGATIVE, A NON-PORTABLE GENERATOR IS SELECTED
C        BASED ON CDC'S RANF AND RANSET FUNCTIONS.
C        IF ISEED=0, GAUSS ALWAYS RETURNS ZERO.
C
C        THIS METHOD HAS BEEN SHOWN TO PRODUCE RELIABLY NORMAL
C        DEVIATES EVEN IN THE TAILS OF THE DISTRIBUTION.
C        ON THE AVERAGE THERE ARE 1.37746 UNIFORM RANDOM NUMBERS
C        NEEDED FOR EACH CALL OF GAUSS, CONSIDERABLY FEWER THAN
```

```
C           THE NUMBER REQUIRED BY OTHER METHODS, INCLUDING THE MORE
C           COMMONLY USED TWELVE-SUM APPROACH.
C
C       METHOD
C           REFERENCE: RICHARD P. BRENT, 'A GAUSSIAN PSEUDO-RANDOM
C           NUMBER GENERATOR', THE COMMUNICATIONS OF THE ACM,
C           DECEMBER 1974.
C
C           THE METHOD USED WAS SUGGESTED BY VON NEUMANN, AND
C           IMPROVED BY FORSYTHE, AHRENS, DIETER AND BRENT.
C
C
C           STEP 1   IF THE FIRST CALL, TAKE A SAMPLE U FROM THE UNIFORM
C                    DISTRIBUTION ON (0,1), OTHERWISE U HAS BEEN SAVED
C                    FROM A PREVIOUS CALL.
C
C           STEP 2   SET X = A(I) + (A(I+1)-A(I))*U AND U(0)=G(X).
C
C           STEP 3   TAKE INDEPENDENT SAMPLES U(1),U(2),.... FROM
C                    THE UNIFORM DISTRIBUTION ON (0,1) UNTIL FOR SOME
C                    K.GE.1, U(K-1).LE.U(K).
C
C           STEP 4   SET U = (U(K)-U(K-1))/(1.-U(K-1))
C
C           STEP 5   IF K IS EVEN GO BACK TO STEP 2, OTHERWISE DETERMINE
C                    SIGN FOR X, ADJUST X TO MATCH THE DESIRED MEAN AND
C                    STANDARD DEVIATION, AND RETURN X, THE NORMAL DEVIATE.
C
C       ...........................................................
C
C
C       WARNINGS
C           DIMENSION AND DATA STATEMENTS BELOW ARE MACHINE DEPENDENT.
C           THE DIMENSION OF D MUST BE AT LEAST THE NUMBER OF BITS IN THE
C           FRACTION OF A FLOATING-POINT NUMBER.  THUS, ON MOST MACHINES,
C           THE DATA STATEMENT BELOW, AND THE SIZE OF D CAN BE TRUNCATED.
C           U MUST BE PRESERVED BETWEEN CALLS.
C
C
C       COMMON / ICOM / ICONT, ISEED, IPASS, IPRRUN, IPGSIZ, IRUN, IDATA
C      +                SEED
        COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
       1             IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
        DIMENSION  D(60)
C
        REAL  MEAN
        DATA  D / 0.674489750, 0.475859630, 0.383771164, 0.328611323,
       +          0.291142827, 0.263684322, 0.242508452, 0.225567444,
       +          0.211634166, 0.199924267, 0.189910758, 0.181225181,
       +          0.173601400, 0.166841909, 0.160796729, 0.155349717,
       +          0.150409384, 0.145902577, 0.141770033, 0.137963174,
       +          0.134441762, 0.131172150, 0.128125965, 0.125279090,
       +          0.122610883, 0.120103560, 0.117741707, 0.115511892,
     +            0.113402349, 0.111402720, 0.109503852, 0.107697617,
     +            0.105976772, 0.104334841, 0.102766012, 0.101265052,
     +            0.099827234, 0.098448282, 0.097124309, 0.095851778,
     +            0.094627461, 0.093448407, 0.092311909, 0.091215482,
     +            0.090156838, 0.089133866, 0.088144619, 0.087187293,
     +            0.086260215, 0.085361834, 0.084490706, 0.083645487,
     +            0.082824924, 0.082027847, 0.081253162, 0.080499844,
     +            0.079766932, 0.079053527, 0.078358781, 0.077681899 /
C
C *** END OF MACHINE-DEPENDENT STATEMENTS
C
      DATA  U / 0. /
      DATA  ISELCT / 1 /
C
      IF  (STD .LT. 0.)  ISELCT = 1
      A = 0.
      I = 0
      GO TO  ( 100, 140, 140, 190 ), ISELCT
C
C *** STEP 1 AND INITIALIZATION:
C *** SET ISELCT SO SUBSEQUENT CALLS GO DIRECTLY TO THE APPROPRIATE
C *** GENERATOR: I.E. IF ISEED<0, =0 OR >0, SELECT THE NON-PORTABLE,
C *** ZERO PRODUCING, OR PORTABLE GENERATORS, RESPECTIVELY.
C *** IF THE CDC NON-PORTABLE CASE IS SELECTED (ISEED<0),
C *** INITIALIZE THIS GENERATOR USING 'RANSET';  FOR THE PORTABLE
C *** GENERATOR, INITIALIZE THE CONSTANTS THAT ARE USED.
C
  100 IF  (ISEED) 110, 120, 130
C
  110 ISELCT = 3
      CALL  RANSET (ISEED)
      U = RANF()
      U = U + U
      IF  (U .GE. 1.)  U = U - 1.
      GO TO  140
C
  120 ISELCT = 4
      GO TO  190
C
  130 ISELCT = 2
      DATA  C7P5 / 16807. / , C2P31M / 2147483647. / , C2P31 /
     +      2147483648. /
      SEED = ISEED
      SEED = AMOD(C7P5*SEED, C2P31M)
      U = SEED / C2P31
      U = U + U
      IF  (U .GE. 1.)  U = U - 1.
C
C *** INCREMENT COUNTER I AND DISPLACEMENT U IF LEADING BIT OF U IS ONE
C
  140 U = U + U
      IF  (U .LT. 1.)  GO TO  150
      U = U - 1.0
      I = I + 1
      A = A + D(I)
      GO TO  140
C
C *** STEP 2:  NOTE THAT U DENOTES UNIFORM SAMPLES U(K), WHERE K IS ODD
C *** V DENOTES THOSE WHERE K IS EVEN.
C *** FORM W UNIFORM ON O. LE. W .LT. D(I+1) FROM U.
C *** W(I+1)=(A(I+1)-A(I))*U
C
  150 W = D(I + 1)*U
C
C *** SET V= U(0)=G(X) WHERE G(X)=.5*(X**2-(A(I))**2)
C
      V = W*(0.5*W + A)
C
C *** STEP 3:  GENERATE NEW UNIFORM U.
C *** IF PORTABLE GENERATOR HAS BEEN SELECTED, UNIFORM SAMPLES
C *** U(I) ARE COMPUTED FROM THIS RECURRENCE RELATION:
C ***      U(I+1) = 7**5 U(I) MODULO(2**31 - 1)
C *** IF NONPORTABLE GENERATOR HAS BEEN SELECTED, RANF PROVIDES
C *** UNIFORM SAMPLES.
C
  160 IF  (ISELCT .EQ. 2)  SEED = AMOD(C7P5*SEED, C2P31M)
      IF  (ISELCT .EQ. 2)  U = SEED / C2P31
      IF  (ISELCT .EQ. 3)  U = RANF()
C
C *** ACCEPT W AS A RANDOM SAMPLE IF V .LE. U.
C
      IF  (V .LE. U)  GO TO  170
C
C *** GENERATE NEW RANDOM V.
C
      IF  (ISELCT .EQ. 2)  SEED = AMOD(C7P5*SEED, C2P31M)
      IF  (ISELCT .EQ. 2)  V = SEED / C2P31
      IF  (ISELCT .EQ. 3)  V = RANF()
C
C *** CONTINUE TAKING UNIFORM RANDOM SAMPLES UNTIL U .LE. V.
C
       IF   (U .GT. V)  GO TO  160
C
C *** STEPS 4 AND 5 FOR K EVEN:  REJECT W AND FORM A NEW
C *** UNIFORM U FROM U AND V.  GO BACK TO STEP 2.
C
      U = (V - U) / (1. - U)
      GO TO  150
C
C *** STEPS 4 AND 5 FOR K ODD:  FORM NEW UNIFORM U FROM U AND V
C *** (TO BE USED IN NEXT CALL).
C
  170 U = (U - V) / (1. - V)
C
C *** USING FIRST BIT OF U TO DETERMINE SIGN, RETURN NORMAL DEVIATE.
C
```

AD-A164 034 PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT 3/5
MANEUVERING TARGET..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R B ANDERSON
UNCLASSIFIED DEC 85 AFIT/GE/ENG/85D-2 F/G 19/5 NL



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS-1963-A

```
      U = U + U
      IF  (U .LT. 1.)  GO TO  180
      U = U - 1.
      GAUSS = (W + A)*STD + MEAN
      RETURN
C
  180 GAUSS =  - (W + A)*STD + MEAN
      RETURN
C
C *** GENERATE ALL ZEROS
C
  190 GAUSS = 0.0
      RETURN
      END
*DECK PROPAG
      SUBROUTINE PROPAG
C
C +++ =PROPAG= IS CALLED FOR EVERY RUN FOR EVERY TIME VALUE
C +++ STORED ON THE EXTERNAL TRAJECTORY OUTPUT OF HILLS.
C +++ PROPAG EMPLOYS A CONSTANT STATE TRANSITION MATRIX
C +++ CALCULATED IN SUBROUTINE INITPRB TO PERFORM THE EXTENDED
C +++ KALMAN FILTER PROPAGATION.  THIS IS POSSIBLE BECAUSE OF
C +++ THE LINEAR DYNAMICS MODEL.  THE EQUATIONS EMPLOYED ARE:
****************************************************************
****************************************************************
*
C +++ (EQ 1)  XF(TI-) = ROTATE[PHI*XF(TI-1+)] + BD*U
C +++ (EQ 2)  P(TI-)  = ROTATE*[PHI*P(TI-1+)*PHIT]*ROTTRAN + QD
*
***************************************************************
***************************************************************
C +++ WHERE ROTATE AND ROTTRAN ARE NECESSARY TO ROTATE XF(TI-1+)
C +++ AND P(TI-1+) INTO THE SAME FRAME AT TIME (TI-).  THIS IS
C +++ NECESSARY BECAUSE HILL5 (WITH OPTION 3 SELECTED) GENEATES
C +++ STATES AND MEASUREMENTS REFERENCED TO A RADAR REFERENCE
C +++ FRAME (A BODY FRAME) WHICH CAN CHANGE DURING THE PROPAGATION.
C +++ NOTE THAT THE BD*U TERM OF EQ 1 IS NOT ROTATED.  THIS IS
C +++ BECAUSE THE BD*U TERM IS REFERENCED TO THE ROTATING FRAME.
C +++ NOTE THAT QD IS NOT ROTATED WITH THE REST OF EQ 2.
C +++ THIS IS BECAUSE QD IS REFERENCED TO THE ROTATING FRAME.
C +++ IF IBUG1=1 THE ROTATE MATRIX IS ECHOED TO TAPE6 FOR IRUN=1.
C +++ IF IBUG2=1 THE XF(TI-) VECTOR IS ECHOED TO TAPE6 FOR IRUN=1.
C +++ IF IBUG3=1 THE P(TI-) MATRIX IS ECHOED TO TAPE6 FOR IRUN=1.
C
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1               IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
      COMMON/PHICOM/PHI(9,9)
      COMMON/UPDATE/ZRES(6,1),XFPLUS(9,1),PPLUS(9,9)
      COMMON/PROPA/XFF(9,1),P(9,9),PMINUS(9,9),XFMINUS(9,1)
      COMMON/QDCOM/QD(9,9)
      COMMON/READEXT/XTRAJ(20)
      COMMON/ROTCOM/ROTATE(9,9)
      COMMON/QFCOM/QFIN(3)
      DIMENSION WRKSPCX(9,1),WRKSPCP(9,9)
C
C *** IF TIME IS ZERO INITIALIZE TURNOLD AND PHHIOLD
C
      IF (T .EQ. 0.) THEN
         TURNOLD=XTRAJ(19)
         PHHIOLD=XTRAJ(20)
         XFPLUS(1,1)=XTRAJ(7)
         XFPLUS(2,1)=XTRAJ(8)
         XFPLUS(3,1)=XTRAJ(9)
         XFPLUS(4,1)=XTRAJ(10)
         XFPLUS(5,1)=XTRAJ(11)
         XFPLUS(6,1)=XTRAJ(12)
         XFPLUS(7,1)=XTRAJ(13)
         XFPLUS(8,1)=XTRAJ(14)
         XFPLUS(9,1)=XTRAJ(15)
         RETURN
      ENDIF
C
C *** ROTATE REORIENTATES THE REFERENCE FRAME AT TIME T(I-1) TO
C *** THAT AT TIME T(I).  THIS IS ACCOMPLISHED BY DIRECTION COSINE
C *** MATRICES.  FIRST, HEADING AND ROLL ANGLES (EXTERNAL TRAJECTORY
C *** VALUES) AT TIME T(I) ARE DIFFERENCED WITH THOSE AT TIME T(I-1)
C *** TO FORM DELTA ANGLES TO USE IN THE DIRECTION COSINE MATRICES.
C *** THE DIRECTION COSINE MATRIX, ROTATE, IS THEN APPLIED TO THE
C *** FILTER'S XF- AND PF-.
C
C *** DETERMINE DELTA ROTATION ANGLES
      DELTURN = XTRAJ(19) - TURNOLD
      DELPHHI  = XTRAJ(20) - PHHIOLD
      TURNOLD = XTRAJ(19)
      PHHIOLD  = XTRAJ(20)
C
C *** FORM DIRECTION COSINE MATRIX, ROTATE
      ROTATE(1,1) = COS(DELTURN)
      ROTATE(2,2) = ROTATE(1,1)
      ROTATE(3,3) = ROTATE(1,1)
      ROTATE(1,4) = SIN(DELTURN)
      ROTATE(2,5) = ROTATE (1,4)
      ROTATE(3,6) = ROTATE (1,4)
      ROTATE(1,7) = 0.
      ROTATE(2,8) = ROTATE(1,7)
      ROTATE(3,9) = ROTATE(1,7)
      ROTATE(4,1) = -COS(DELPHHI)*SIN(DELTURN)
      ROTATE(5,2) = ROTATE(4,1)
      ROTATE(6,3) = ROTATE(4,1)
      ROTATE(4,4) = COS(DELPHHI)*COS(DELTURN)
      ROTATE(5,5) = ROTATE(4,4)
      ROTATE(6,6) = ROTATE(4,4)
      ROTATE(4,7) = SIN(DELPHHI)
      ROTATE(5,8) = ROTATE(4,7)
      ROTATE(6,9) = ROTATE(4,7)
      ROTATE(7,1) = SIN(DELPHHI)*SIN(DELTURN)
      ROTATE(8,2) = ROTATE(7,1)
      ROTATE(9,3) = ROTATE(7,1)
      ROTATE(7,4) = -SIN(DELPHHI)*COS(DELTURN)
      ROTATE(8,5) = ROTATE(7,4)
      ROTATE(9,6) = ROTATE(7,4)
      ROTATE(7,7) = COS(DELPHHI)
      ROTATE(8,8) = ROTATE(7,7)
      ROTATE(9,9) = ROTATE(7,7)
C
C *** NOW PERFORM EQ 1 MATRIX MULTIPLICATION CALCULATIONS
C *** X(TI-)=XFMINUS
C
      CALL MMUL(PHI,XFPLUS,XFF,WRKSPCX,NF,NF,1)
      CALL MMUL(ROTATE,XFF,XFMINUS,WRKSPCX,NF,NF,1)
C
C *** NOW ADD EQ1 BU TERM TO GET XFMINUS
C
C     SINCE VFX(OR XTRAJ(16)) IS THE ONLY NONZERO BD*U TERM
C     SUBTRACT BD*VFX FROM XFMINUS(1,1)
      XFMINUS(1,1)=XFMINUS(1,1)-DTMEAS*XTRAJ(16)
C
C *** NOW PERFORM EQ 2 CALCULATIONS, P(TI-)=PMINUS
C     FIRST TERM (RIGHT HAND TERM BEFORE + SIGN)
      CALL MABT(PPLUS,PHI,P,WRKSPCP,NF,NF,NF)
      CALL MMUL(PHI,P,P,WRKSPCP,NF,NF,NF)
      CALL MABT(P,ROTATE,P,WRKSPCP,NF,NF,NF)
      CALL MMUL(ROTATE,P,P,WRKSPCP,NF,NF,NF)
C
C     NOW ALLOW FOR 'ADAPTIVE TYPE' CHANGES
C
      IF (T .GE. 3.00 .AND. T .LT. 3.10) THEN
          QFIN(1)=37325.
          QFIN(2)=37325.
          QFIN(3)=37325.
          CALL ADAPQD
      ENDIF
      IF(T .GE. 6.00 .AND. T .LT. 6.10) THEN
          QFIN(1)=373250.
          QFIN(2)=373250.
          QFIN(3)=373250.
          CALL ADAPQD
      ENDIF
C
C     NOW ADD QD TO P TO GET P(TI-)=PMINUS
      CALL MADD(P,QD,PMINUS,9,9)
C
C *** TAPE6 OUTPUT
C
      WRITE(6,*)
      WRITE(6,*)'************************************************'
      WRITE(6,*)'****************TIME= ',T,' ********************'
      IF (IBUG1 .EQ. 1 .AND. IRUN .EQ. 1) THEN
         WRITE(6,*)
         WRITE(6,*)'ROTATE MATRIX AT TIME = ',T
         WRITE(6,*)
         DO 110 K=1,NF
            WRITE(6,200)(K,I,ROTATE(K,I),I=1,NF)
  110    CONTINUE
         WRITE(6,*)
      ENDIF
      IF (IBUG2 .EQ. 1 .AND. IRUN .EQ. 1) THEN
         WRITE(6,*)
         WRITE(6,*)'XF(TI-) OR XFMINUS VECTOR AT TIME = ',T
         WRITE(6,*)
         WRITE(6,210)(I,XFMINUS(I,1),I=1,NF)
         WRITE(6,*)
      ENDIF
      IF (IBUG3 .EQ. 1 .AND. IRUN .EQ. 1) THEN
         WRITE(6,*)
         WRITE(6,*)'P(TI-) OR PMINUS MATRIX AT TIME = ',T
         WRITE(6,*)
         DO 120 K=1,NF
            WRITE(6,200)(K,I,PMINUS(K,I),I=1,NF)
  120    CONTINUE
         WRITE(6,*)
      ENDIF
      RETURN
  200 FORMAT(9(I1,I1,E11.5,1X))
  210 FORMAT(9(1X,I1,E11.5,1X))
      END
*DECK ADAPQD
      SUBROUTINE ADAPQD
C +++ =ADAPQD= ALLOWS FOR FORCING ADAPTIVE CHANGES IN Q BASED
C +++ ON LOOKING AT PREVIOUS PLOTS AND Q TO OBTAIN DESIRED
C +++ PERFORMANCE.
      COMMON/QFCOM/QFIN(3)
      COMMON/QDCOM/QD(9,9)
      COMMON/TAUCOM/TAU(3)
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1                IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
C
      QD(1,1)=((QFIN(1)*TAU(1)**5)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(1)))+
     1       2.*DTMEAS/TAU(1)+2.*DTMEAS**3/(3.*TAU(1)**3)-
     2       2.*DTMEAS**2/TAU(1)**2-(4.*DTMEAS/TAU(1))*
     3       EXP(-(DTMEAS/TAU(1))))
      QD(1,2)=((QFIN(1)*TAU(1)**4)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(1)))-
     1       2.*EXP(-(DTMEAS/TAU(1)))+(2.*DTMEAS/TAU(1))*
     2       EXP(-(DTMEAS/TAU(1)))-2.*DTMEAS/TAU(1)+
     3       DTMEAS**2/TAU(1)**2)
      QD(1,3)=((QFIN(1)*TAU(1)**3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(1)))-
     1       (2.*DTMEAS/TAU(1))*EXP(-(DTMEAS/TAU(1))))
      QD(2,2)=((QFIN(1)*TAU(1)**3)/2.)*(4.*EXP(-(DTMEAS/TAU(1)))-3.-
     1       EXP(-(2.*DTMEAS/TAU(1)))+2.*DTMEAS/TAU(1))
      QD(2,3)=((QFIN(1)*TAU(1)**2)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(1)))-
     1       2.*EXP(-(DTMEAS/TAU(1))))
      QD(3,3)=(QFIN(1)*TAU(1)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(1))))
      QD(2,1)=QD(1,2)
      QD(3,1)=QD(1,3)
      QD(3,2)=QD(2,3)
      QD(4,4)=((QFIN(2)*TAU(2)**5)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(2)))+
     1        2.*DTMEAS/TAU(2)+2.*DTMEAS**3/(3.*TAU(2)**3)-
     2        2.*DTMEAS**2/TAU(2)**2-(4.*DTMEAS/TAU(2))*
     3        EXP(-(DTMEAS/TAU(2))))
      QD(4,5)=((QFIN(2)*TAU(2)**4)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(2)))-
     1        2.*EXP(-(DTMEAS/TAU(2)))+(2.*DTMEAS/TAU(2))*
     2        EXP(-(DTMEAS/TAU(2)))-2.*DTMEAS/TAU(2)+
     3        DTMEAS**2/TAU(2)**2)
      QD(4,6)=((QFIN(2)*TAU(2)**3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(2)))-
     1        (2.*DTMEAS/TAU(2))*EXP(-(DTMEAS/TAU(2))))
      QD(5,5)=((QFIN(2)*TAU(2)**3)/2.)*(4.*EXP(-(DTMEAS/TAU(2)))-3.-
     1        EXP(-(2.*DTMEAS/TAU(2)))+2.*DTMEAS/TAU(2))
      QD(5,6)=((QFIN(2)*TAU(2)**2)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(2)))-
     1        2.*EXP(-(DTMEAS/TAU(2))))
      QD(6,6)=(QFIN(2)*TAU(2)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(2))))
      QD(5,4)=QD(4,5)
      QD(6,4)=QD(4,6)
      QD(6,5)=QD(5,6)
      QD(7,7)=((QFIN(3)*TAU(3)**5)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(3)))+
     1        2.*DTMEAS/TAU(3)+2.*DTMEAS**3/(3.*TAU(3)**3)-
     2        2.*DTMEAS**2/TAU(3)**2-(4.*DTMEAS/TAU(3))*
     3        EXP(-(DTMEAS/TAU(3))))
      QD(7,8)=((QFIN(3)*TAU(3)**4)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(3)))-
     1        2.*EXP(-(DTMEAS/TAU(3)))+(2.*DTMEAS/TAU(3))*
     2        EXP(-(DTMEAS/TAU(3)))-2.*DTMEAS/TAU(3)+
     3        DTMEAS**2/TAU(3)**2)
      QD(7,9)=((QFIN(3)*TAU(3)**3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(3)))-
     1        (2.*DTMEAS/TAU(3))*EXP(-(DTMEAS/TAU(3))))
      QD(8,8)=((QFIN(3)*TAU(3)**3)/2.)*(4.*EXP(-(DTMEAS/TAU(3)))-3.-
     1        EXP(-(2.*DTMEAS/TAU(3)))+2.*DTMEAS/TAU(3))
      QD(8,9)=((QFIN(3)*TAU(3)**2)/2.)*(1.+EXP(-(2.*DTMEAS/TAU(3)))-
     1        2.*EXP(-(DTMEAS/TAU(3))))
      QD(9,9)=(QFIN(3)*TAU(3)/2.)*(1.-EXP(-(2.*DTMEAS/TAU(3))))
      QD(8,7)=QD(7,8)
      QD(9,7)=QD(7,9)
      QD(9,8)=QD(8,9)
      RETURN
      END
*DECK UPDATE
          SUBROUTINE UPDATE
C
C +++ =UPDATE= IS CALLED FOR EVERY RUN FOR EVERY TIME VALUE
C +++ STORED ON THE EXTERNAL TRAJECTORY OUTPUT OF HILL5.
C +++ UPDATE INCORPORATES MEASUREMENT INFORMATION INTO STATE
C +++ AND COVARIANCE ESTIMATES ACCORDING TO THE FOLLOWING:
************************************************************************
************************************************************************
*
C +++ (EQ 3)  K(TI)    = P(TI-)*HT*INV[(H(XF(TI-))*P(TI-)*HT) + R]
C +++ (EQ 4)  XF(TI+) = XF(TI-) + K(TI)*[ZM - ZHM]
```

```
C +++ (EQ 5)  P(TI+)  = P(TI-) - [K(TI)*H(XF(TI-))*P(TI-)]
*
*******************************************************************
*******************************************************************
C +++ WHERE : ZM IS THE "ACTUAL MEASUREMENT REALIZATION."  NOISE
C +++         IS ADDED THROUGH REAL FUNCTION GAUSS.
C +++       : ZHM IS H*X(TI-) OR THE KF ESTIMATE OF THE MEASUREMENT
C +++       : H(XF(TI-)) IS THE PARTIAL DERIVATIVE EVALUATION OF
C +++         THE H MATRIX
C +++ IF IBUG4=1 THE H MATRIX IS ECHOED TO TAPE6 FOR IRUN=1.
C +++ IF IBUG5=1 THE K(TI) VECTOR IS ECHOED TO TAPE6 FOR IRUN=1.
C +++ IF IBUG6=1 THE X(TI+) VECTOR AND P(TI+) MATRIX IS ECHOED TO TAPE6
C +++ FOR IRUN=1.
C
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED.IPASS,IRUN,
     1               IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
      COMMON/UPDATE/ZRES(6,1),XFPLUS(9,1),PPLUS(9,9)
      COMMON/RFCOM/RMAT(6,6)
      COMMON/PROPA/XFF(9,1),P(9,9),PMINUS(9,9),XFMINUS(9,1)
      COMMON/SDEVCOM/STD(6)
      COMMON/READEXT/XTRAJ(20)
      COMMON/HCOM/H(6,9)
      DIMENSION TEMP1(6,9),TEMP2(6,6),TEMPK(9,6),WRKSPC1(6,9),
     1          WRKSPC2(6,6),WRKSPC3(9,6),WRKSPC4(9,1),
     2          WRKSPC5(9,9),WRKVEC1(6),WRKVEC2(6)
      REAL XF(9),H
C
C *** IF TIME IS ZERO RETURN
C
      IF (T .EQ. 0.) RETURN
C
C *** FIRST, CHANGE FROM DOUBLE TO SINGLE SUBSCRIPT
C
      XF(1)=XFMINUS(1,1)
      XF(2)=XFMINUS(2,1)
      XF(3)=XFMINUS(3,1)
      XF(4)=XFMINUS(4,1)
      XF(5)=XFMINUS(5,1)
      XF(6)=XFMINUS(6,1)
      XF(7)=XFMINUS(7,1)
      XF(8)=XFMINUS(8,1)
      XF(9)=XFMINUS(9,1)
C
C *** NOW CALCULATE 6*9 H MATRIX
C
C     CALCULATE SOME INTERIM VALUES FOR EFFICIENCY
C
         SUMSQ = XF(1)**2 + XF(4)**2 + XF(7)**2
         SUMSQ12= SQRT(SUMSQ)
         SUMSQ32= (SUMSQ)**1.5
         SUMSQ2 = (SUMSQ)**2
         PSUM   = XF(1)**2 + XF(4)**2
         PSUM12 = SQRT(PSUM)
         PSUM2  = PSUM**2
         PSUM32 = PSUM**1.5
         ADD1   = (XF(1)*(XF(2)-XTRAJ(16)))+XF(4)*XF(5) + XF(7)*XF(8)
         ADD2   = XF(4)**2 - XF(1)**2
         ADD3   = XF(8)*PSUM - XF(7)*(XF(1)*(XF(2)-XTRAJ(16)) +
     1  XF(4)*XF(5))
C
C      CONTINUE CALCULATING H
         H(1,1)  = XF(1)/SUMSQ12
         H(1,4)  = XF(4)/SUMSQ12
         H(1,7)  = XF(7)/SUMSQ12
         H(2,1)  = ((XF(2)-XTRAJ(16))*SUMSQ - XF(1)*ADD1)/SUMSQ32
         H(2,2)  = H(1,1)
         H(2,4)  = (XF(5)*SUMSQ - XF(4)*ADD1)/SUMSQ32
         H(2,5)  = H(1,4)
         H(2,7)  = (XF(8)*SUMSQ - XF(7)*ADD1)/SUMSQ32
         H(2,8)  = H(1,7)
         H(3,1) = -XF(4)/PSUM
         H(3,4) = XF(1)/PSUM
         H(4,1) = (XF(1)*XF(7))/(SUMSQ*PSUM12)
         H(4,4) = (XF(4)*XF(7))/(SUMSQ*PSUM12)
         H(4,7) = -PSUM12/SUMSQ
         H(5,1) = (XF(5)*ADD2 + 2.*XF(1)*(XF(2)-XTRAJ(16))*XF(4))
     1            /PSUM2
         H(5,2) = H(3,1)
         H(5,4) = ((XF(2)-XTRAJ(16))*ADD2 - 2.*XF(1)*XF(4)*XF(5))
     1            /PSUM2
         H(5,5) = H(3,4)
         H(6,1) =-((SUMSQ*PSUM*(2.*XF(1)*XF(8) -
     1            (XF(2)-XTRAJ(16))*XF(7))) - ADD3*
     2            XF(1)*(2.*PSUM + SUMSQ))/(SUMSQ2*PSUM32)
         H(6,2) = H(4,1)
         H(6,4) =-((SUMSQ*PSUM*(2.*XF(4)*XF(8) - XF(5)*XF(7))) - ADD3*
     1            XF(4)*(     SUM + SUMSQ))/(SUMSQ2*PSUM32)
         H(6,5) = H(4,4)
         H(6,7) = ((SUMSQ*PSUM*(XF(1)*(XF(2)-XTRAJ(16)) +
     1            XF(4)*XF(5)) + ADD3*
     2            2.*XF(7)*PSUM))/(SUMSQ2*PSUM32)
         H(6,8) = -PSUM/(SUMSQ*PSUM12)
C
C*** NOW CALCULATE EQ 3, K(TI)=TEMPK
C
      CALL MMUL(H,PMINUS,TEMP1,WRKSPC1,6,9,9)
      CALL MABT(TEMP1,H,TEMP2,WRKSPC2,6,9,6)
      CALL MADD(TEMP2,RMAT,TEMP2,6,6)
      CALL MINV(TEMP2,6,DET,WRKVEC1,WRKVEC2)
      CALL MATB(H,TEMP2,TEMPK,WRKSPC3,9,6,6)
      CALL MMUL(PMINUS,TEMPK,TEMPK,WRKSPC3,9,9,6)
C
C *** NOW CLACULATE EQ 4, XF(TI+)=XFPLUS
C
C      FIRST FORM [ZM-ZHM]
      ZRES(1,1) = XTRAJ(1) - SUMSQ12 + GAUSS(0.,STD(1))
      ZRES(2,1) = XTRAJ(2) - ADD1/SUMSQ12 + GAUSS(0.,STD(2))
      ZRES(3,1) = XTRAJ(3) -ATAN(XF(4)/XF(1)) + GAUSS(0.,STD(3))
      ZRES(4,1) = XTRAJ(4) + ATAN(XF(7)/PSUM12) + GAUSS(0.,STD(4))
      ZRES(5,1) = -(XF(1)*XF(5) - (XF(2)-XTRAJ(16))*XF(4))/PSUM
     1        +XTRAJ(5) + GAUSS(0.,STD(5))
      ZRES(6,1) = (XF(8)*PSUM - XF(7)*(XF(1)*(XF(2)-XTRAJ(16))) +
     1          XF(4)*XF(5))/
     2          (SUMSQ*PSUM12) + GAUSS(0.,STD(6)) + XTRAJ(6)
C
C     NOW FINISH EQ 4
      CALL MMUL(TEMPK,ZRES,XFPLUS,WRKSPC4,9,6,1)
      CALL MADD(XFPLUS,XFMINUS,XFPLUS,9,1)
C
C *** NOW CALCULATE EQ 5, P(TI+)=PPLUS
C
      CALL MMUL(H,PMINUS,TEMP1,WRKSPC1,6,9,9)
      CALL MMUL(TEMPK,TEMP1,PPLUS,WRKSPC5,9,6,9)
      CALL MSUB(PMINUS,PPLUS,PPLUS,9,9)
C
C *** TAPE6 OUTPUT
C
      IF (IBUG4 .EQ. 1 .AND. IRUN .EQ. 1) THEN
         WRITE(6,*)
         WRITE(6,*)'H MATRIX AT TIME = ',T
         WRITE(6,*)
         DO 220 K=1,M
            WRITE(6,300)(K,I,H(K,I),I=1,NF)
  220    CONTINUE
         WRITE(6,*)
      ENDIF
      IF (IBUG5 .EQ. 1 .AND. IRUN .EQ. 1) THEN
         WRITE(6,*)
         WRITE(6,*)'K(TI) OR K MATRIX AT TIME = ',T
         WRITE(6,*)
         DO 230 K=1,NF
            WRITE(6,310)(K,I,TEMPK(K,I),I=1,M)
  230    CONTINUE
         WRITE(6,*)
      ENDIF
      IF (IBUG6 .EQ. 1 .AND. IRUN .EQ. 1) THEN
         WRITE(6,*)
         WRITE(6,*)'XF(TI+) OR XFPLUS VECTOR AT TIME = ',T
         WRITE(6,*)
         WRITE(6,320)(I,XFPLUS(I,1),I=1,NF)
         WRITE(6,*)
         WRITE(6,*)
         WRITE(6,*)'P(TI+) OR PPLUS MATRIX AT TIME = ',T
         WRITE(6,*)
         DO 240 K=1,NF
            WRITE(6,300)(K,I,PPLUS(K,I),I=1,NF)
  240    CONTINUE
         WRITE(6,*)
      ENDIF
      RETURN
  300 FORMAT(9(I1,I1,E11.5,1X))
  310 FORMAT(6(I1,I1,E11.5,1X))
  320 FORMAT(9(I1,1X,E11.5,1X))
      END
*DECK SAVSTAT
      SUBROUTINE SAVSTAT
C
C
C +++ =SAVSTAT= IS CALLED FOR EVERY RUN FOR EVERY TIME VALUE
C +++ STORED ON THE EXTERNAL TRAJECTORY OUTPUT OF HILL5.
C +++ SAVSTAT HAS TWO FUNCTIONS.
C *** FIRST, IT SAVES DATA TO BE POSTPROCESSED BY SUBROUTINE
C *** POSTPRC.  SECOND, IT CONTROLS OUTPUT TO TAPE6.
C
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1               IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
      COMMON/READEXT/XTRAJ(20)
      COMMON/UPDATE/ZRES(6,1),XFPLUS(9,1),PPLUS(9,9)
      COMMON/SAVSTAT/XE(9),SUMXE(500,9,2),SUMPPL(500,9)
      COMMON/INITRUN/ICOUNT
C
C *** FORM STATISICAL DATA
C
      ICOUNT=ICOUNT + 1
      DO 500 I=1,NF
         XE(I)=XTRAJ(I+6)-XFPLUS(I,1)
         SUMXE(ICOUNT,I,1)=SUMXE(ICOUNT,I,1) + XE(I)
         SUMXE(ICOUNT,I,2)=SUMXE(ICOUNT,I,2) + XE(I)**2
         SUMPPL(ICOUNT,I) = SUMPPL(ICOUNT,I) + PPLUS(I,I)
  500 CONTINUE
C
C *** PRINT TRUTH STATES, ESTIMATED STATES, ETC, FOR FIRST IPASS
C *** AND LAST IPASS (IF IBUG1 SET EQUAL TO 1).
C
      IF (IBUG7 .EQ. 1) THEN
         IF (IRUN .EQ. 1 .OR. IRUN .EQ. IPASS) THEN
            WRITE(6,*) ' TRUTH STATES XS AT T=  ',T
            WRITE(6,1060)(I,XTRAJ(I+6),I=1,NS)
            WRITE(6,*) 'FILTER STATES XF AT T=  ',T
            WRITE(6,1060)(I,XFPLUS(I,1),I=1,NF)
            WRITE(6,*) 'ERROR:  '
            WRITE(6,1070)(XE(I),I=1,NF)
            WRITE(6,*) 'FILTER VARIANCE SUMS'
            WRITE(6,1070)(SUMPPL(ICOUNT,I),I=1,NF)
            WRITE(6,*)
            WRITE(6,*)
         ENDIF
      ENDIF
      RETURN
C1000 FORMAT(T5, 'TRUTH  STATE ',A4, ' AT T = ', G12.5)
C1010 FORMAT(T5, 'FILTER STATE ',A4, ' AT T = ', G12.5)
C1050 FORMAT((5(6X, I3, '.', 1PG14.6)))
```

```
 1060 FORMAT(9(I1,1X,E11.5,1X))
 1070 FORMAT(9(2X,E11.5,1X))
      END
*DECK POSTPRC
      SUBROUTINE POSTPRC
C
C +++ =POSTPRC= IS CALLED ONCE AT THE ENDING OF A PROBLEM TO
C +++ GENERATE, SAVE, AND PLOT DATA.  THE AVERAGE MEAN OF E, THE
C +++ STANDARD DEVIATION SE OF THE AVERAGE MEAN, AND THE PLUS
C +++ OR MINUS SQUARE ROOT OF THE DIAGONAL OF THE FILTER
C +++ COMPUTED COVARIANCE ARE CALCULATED, SAVED, AND PLOTTED.
C
      COMMON/SAVSTAT/XE(9),SUMXE(500,9,2),SUMPPL(500,9)
      COMMON/INITPRB/NF,NS,M,NXTJ,T,TO,TF,DTMEAS,ISEED,IPASS,IRUN,
     1              IBUG1,IBUG2,IBUG3,IBUG4,IBUG5,IBUG6,IBUG7
      REAL MNE,MNESQ,FILSIG,FILSAG
      INTEGER FUNIT
C
C *** SETUP OUTPUT FILES FOR 'PLOT M'
C
      OPEN(UNIT=11,FILE='PLOTX1',STATUS='NEW')
      OPEN(UNIT=12,FILE='PLOTX2',STATUS='NEW')
      OPEN(UNIT=13,FILE='PLOTX3',STATUS='NEW')
      OPEN(UNIT=14,FILE='PLOTX4',STATUS='NEW')
      OPEN(UNIT=15,FILE='PLOTX5',STATUS='NEW')
      OPEN(UNIT=16,FILE='PLOTX6',STATUS='NEW')
      OPEN(UNIT=17,FILE='PLOTX7',STATUS='NEW')
      OPEN(UNIT=18,FILE='PLOTX8',STATUS='NEW')
      OPEN(UNIT=19,FILE='PLOTX9',STATUS='NEW')
C
C *** REWIND ALL UNITS ABOVE
C
      DO 10 I=11,19
         REWIND I
   10 CONTINUE
C
C *** WRITE POSTPROCESSOR DATA TO THE UNITS
C
      DO 30 I=1,TF/DTMEAS
         T=(I*DTMEAS)-DTMEAS
         DO 20 J=1,NF
            FUNIT = J + 10
            MNE   =SUMXE(I,J,1)/IPASS
            MNESQ =SUMXE(I,J,2)/IPASS
            TRUSIG=MNE + (MNESQ-MNE**2)**.5
            TRUSAG=MNE - (MNESQ-MNE**2)**.5
            FILSIG=SQRT(SUMPPL(I,J)/IPASS)
            FILSAG=-FILSIG
          WRITE(FUNIT,*)T,MNE,TRUSIG,TRUSAG,FILSIG,FILSAG
   20    CONTINUE
   30 CONTINUE
      RETURN
      END
```

```
$PRBINIT
QFIN(1)=373250.,373250.,373250.,
RFIN(1)=2500.,625.,3.00E-04,3.00E-04,4.900E-04,4.900E-04,
TAU(1)=.142857143,.142857143,.142857143,IPASS=5,IBUG1=1,IBUG2=1,IBUG3=1
TF=15.0,$
EOI ENCOUNTERED.
```

APPENDIX F

Filter Operating Time

The operating time of the Kalman filter is estimated by calculating the number of additions, multiplications, divides and square root functions for one filter cycle. Then the function times provided by OO-ALC for the F-4E/G fire control computer are multiplied times the above functions. The number of functions are calculated by formulas (5:403) and then adding the number of calculations required in the evaluation of $\underline{H}$ (see Equation (3-15)). The overall calculation is a worst-case evaluation, since in the evaluation of $\underline{H}$, many of the relations are common and need only be calculated once and then stored for recall. The calculation results that follow do not take this into account. The number of functions required for linear filters are listed in Table F-1. Then the number of functions required in the evaluation of $\underline{H}$ are listed in Table F-2. Table F-3 lists approximate filter cycle times for various filter combinations, calculated by adding entries from Tables F-1 and F-2 and multiplying by corresponding function times.

Table F-1

Linear Filter Update Operations

	Conventional[1] Kalman	U-D[2]	U-D[3]
Adds	2313	2718	2448
Multiplies	2673	2997	2745
Divides	6	62	44
Square Roots	0	0	0

[1]- nine states, six measurements, and nine-by-nine $\underline{Q}_d$
[2]- nine states, six measurements, and nine-by-nine $\underline{Q}_d$
[3]- nine states, four measurements, and nine-by-nine $\underline{Q}_d$

Table F-2

Evaluation of $\underline{H}$ Operations[1]

	Six Measurements	Four Measurements
Adds	133	53
Multiplies	209	77
Divides	24	14
Square Roots	18	12

[1]- worst case design, many of the relations are common and can be stored and recalled, instead of recalculated

Table F-3

Approximate Operations/Operating Times for One Filter Cycle

	Adds	Multiplies	Divides	Square Roots	Time[1] (msec)
Conventional Kalman	2446	2882	30	18	35.78
U-D (6 measurements)	2851	3206	86	18	41.17
U-D (4 measurements)	2501	2822	58	12	35.21

[1] - computed as [(adds)(.003) + (multiplies)(.00868) + (divides)(.02433) + (square roots)(.150)]

Appendix G

Data Plots

SOFEPL DATE AND TIME - 85/09/30. 1[illegible] SOFE DATE AND TIME - 85/09/30. 18.47.15. WPAFB, [illegible]

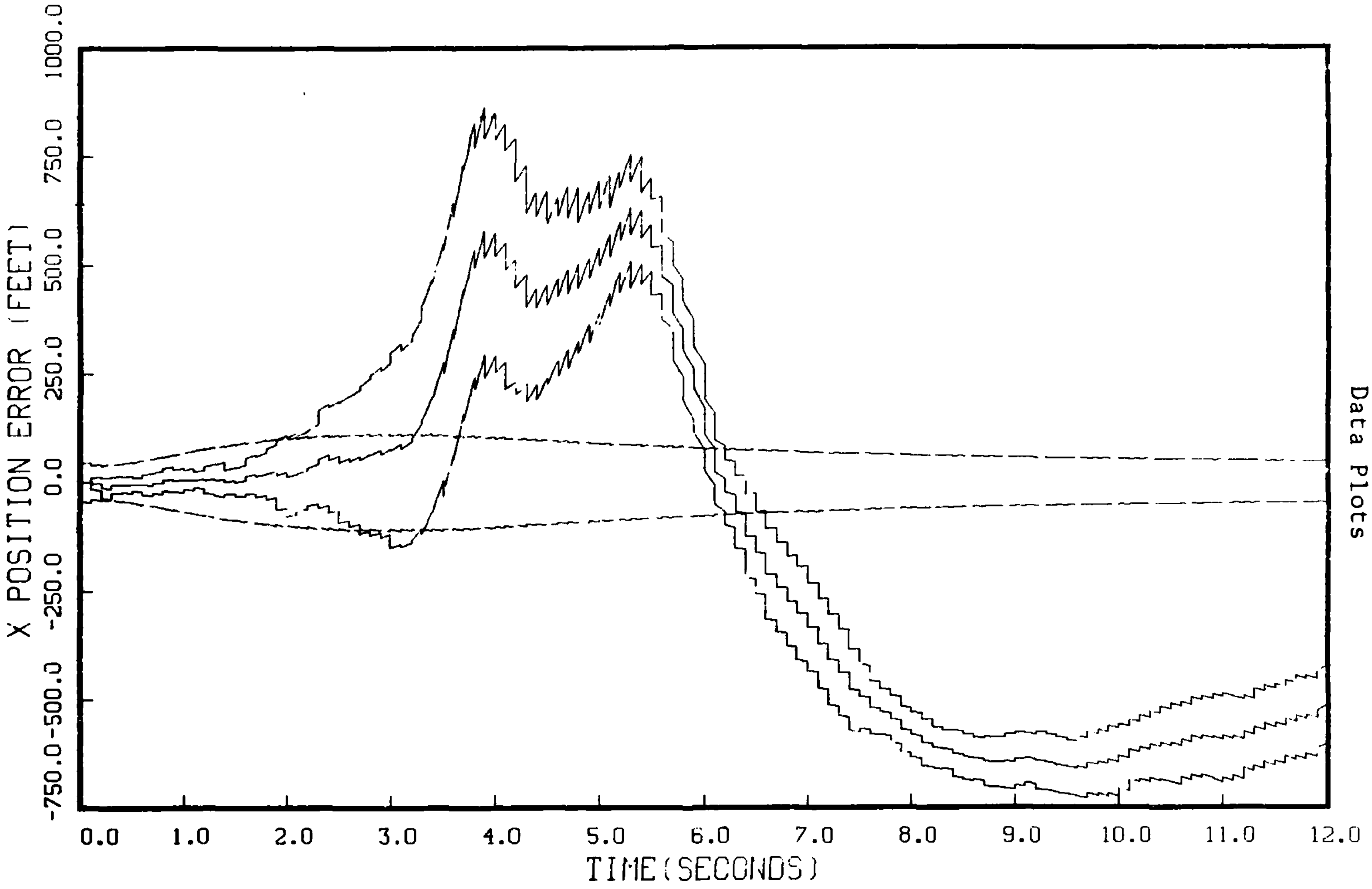


Figure G.1.1.a
STATE 1, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

2. SOFEPL DATE AND TIME - 85/09/30. 19.33.44. SOFE DATE AND TIME - 85/09/30. 18.47.55. WPAFB, DISSPLA, SOFEPL VERSION 2.2

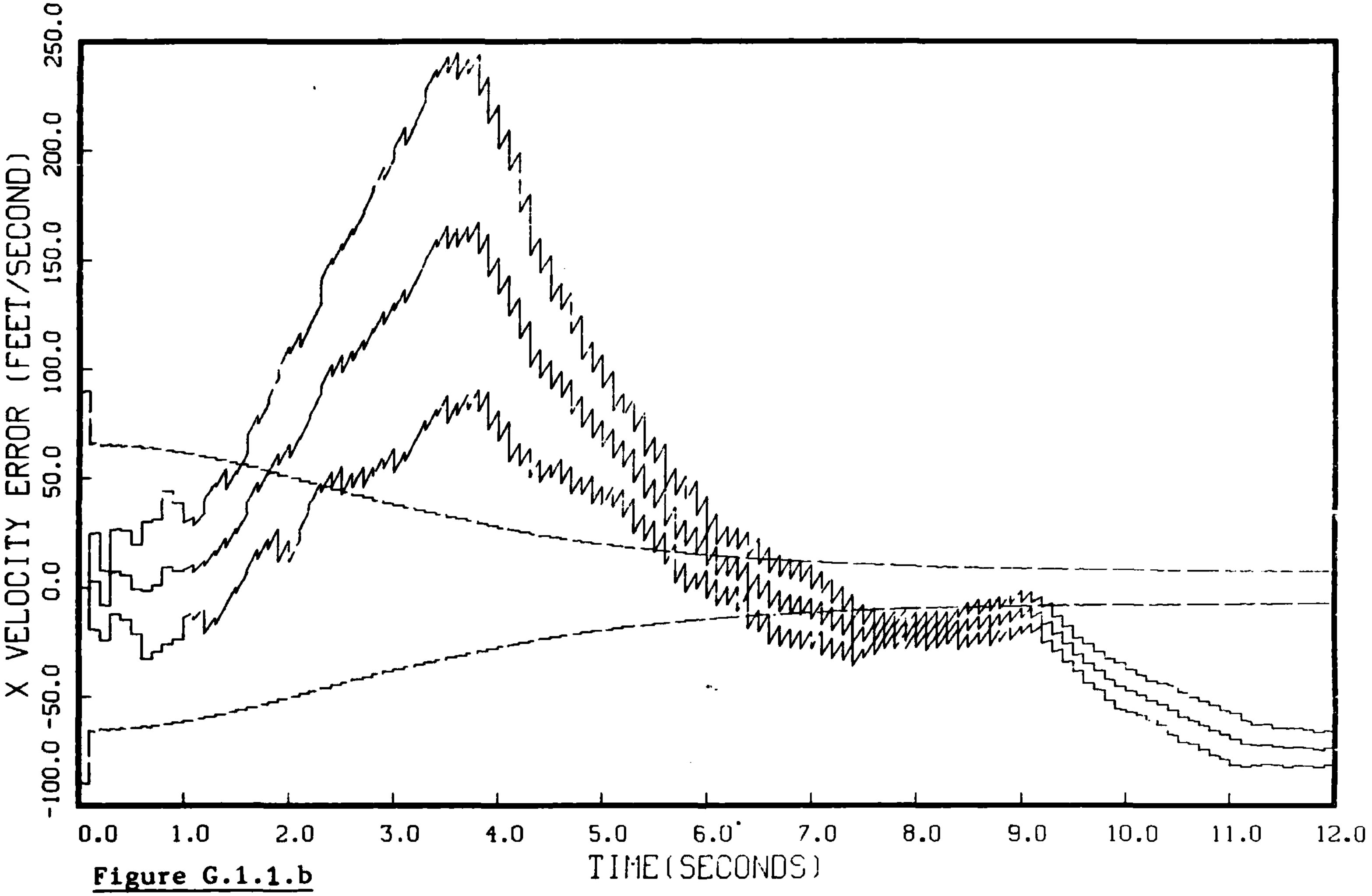


Figure G.1.1.b

STATE 2, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

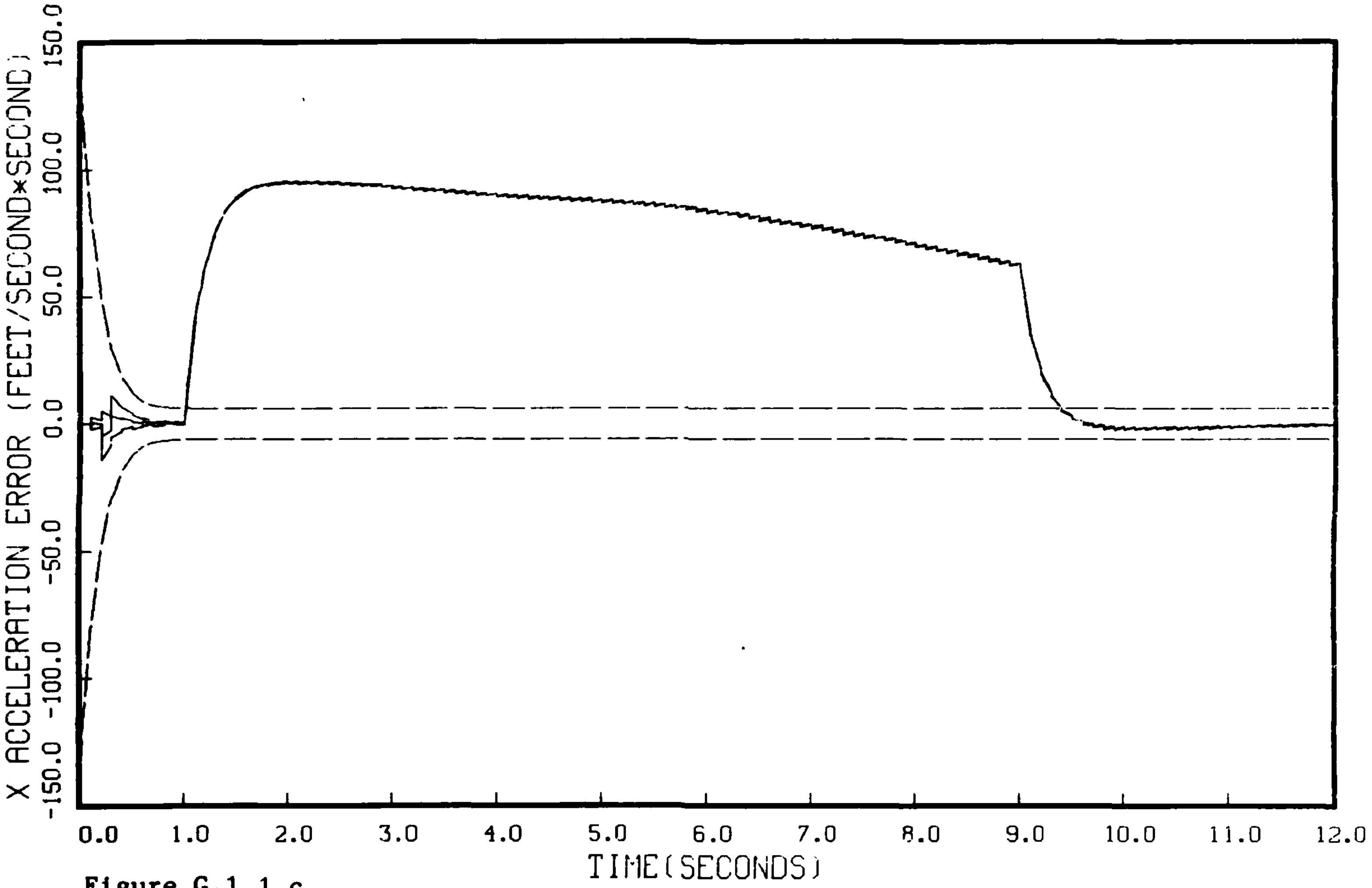


Figure G.1.1.c

STATE 3, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

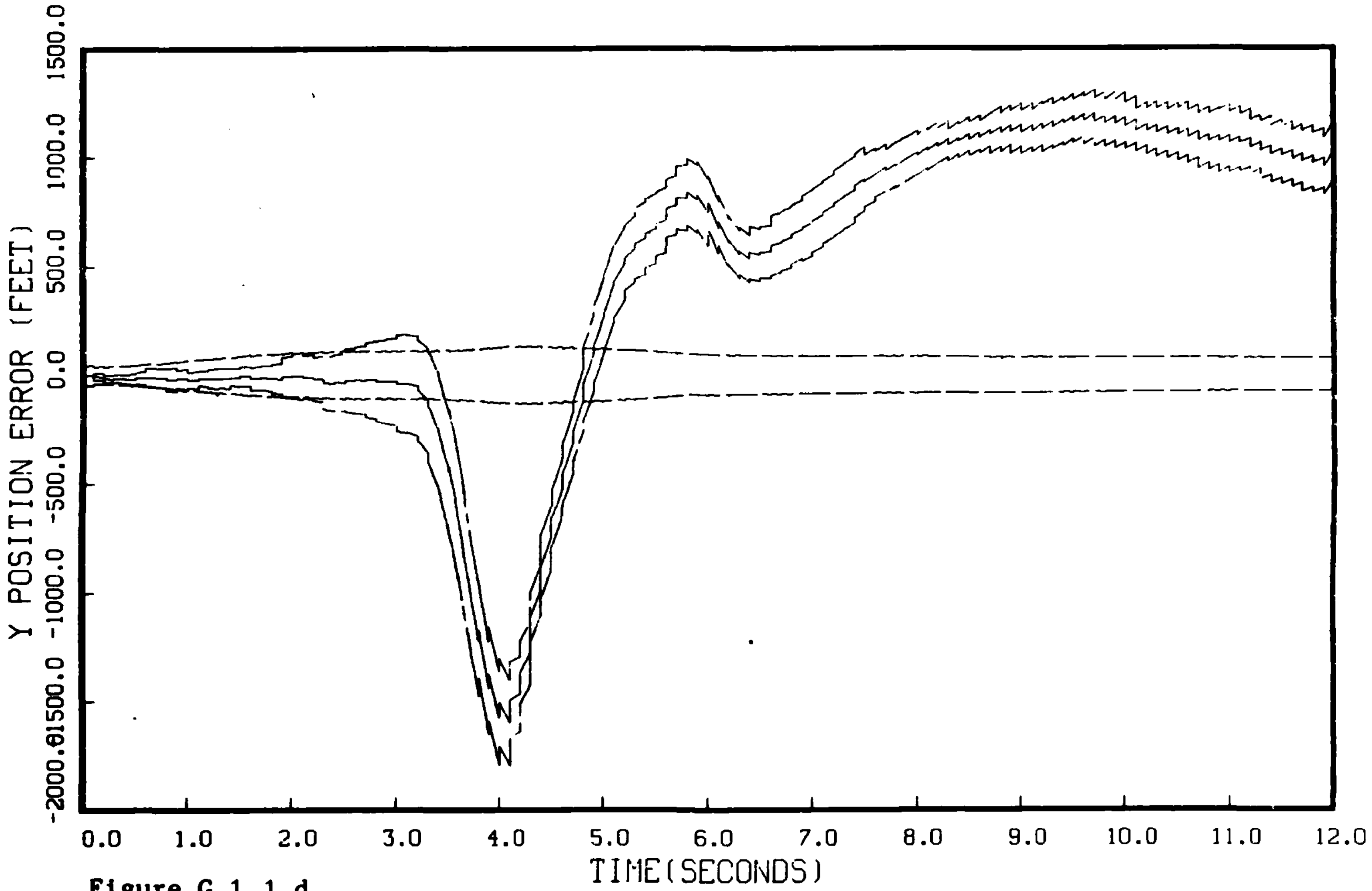


Figure G.1.1.d

STATE 4, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.' , UPDATE-.1, 5 RUNS

SOFEPL DATE AND TIME - 85/09/30. 19.33.44.   SOFE DATE AND TIME - 85/09/30. 18.47.55.   WPAFB, DISSPLA, SOFEPL VERSION 2.2

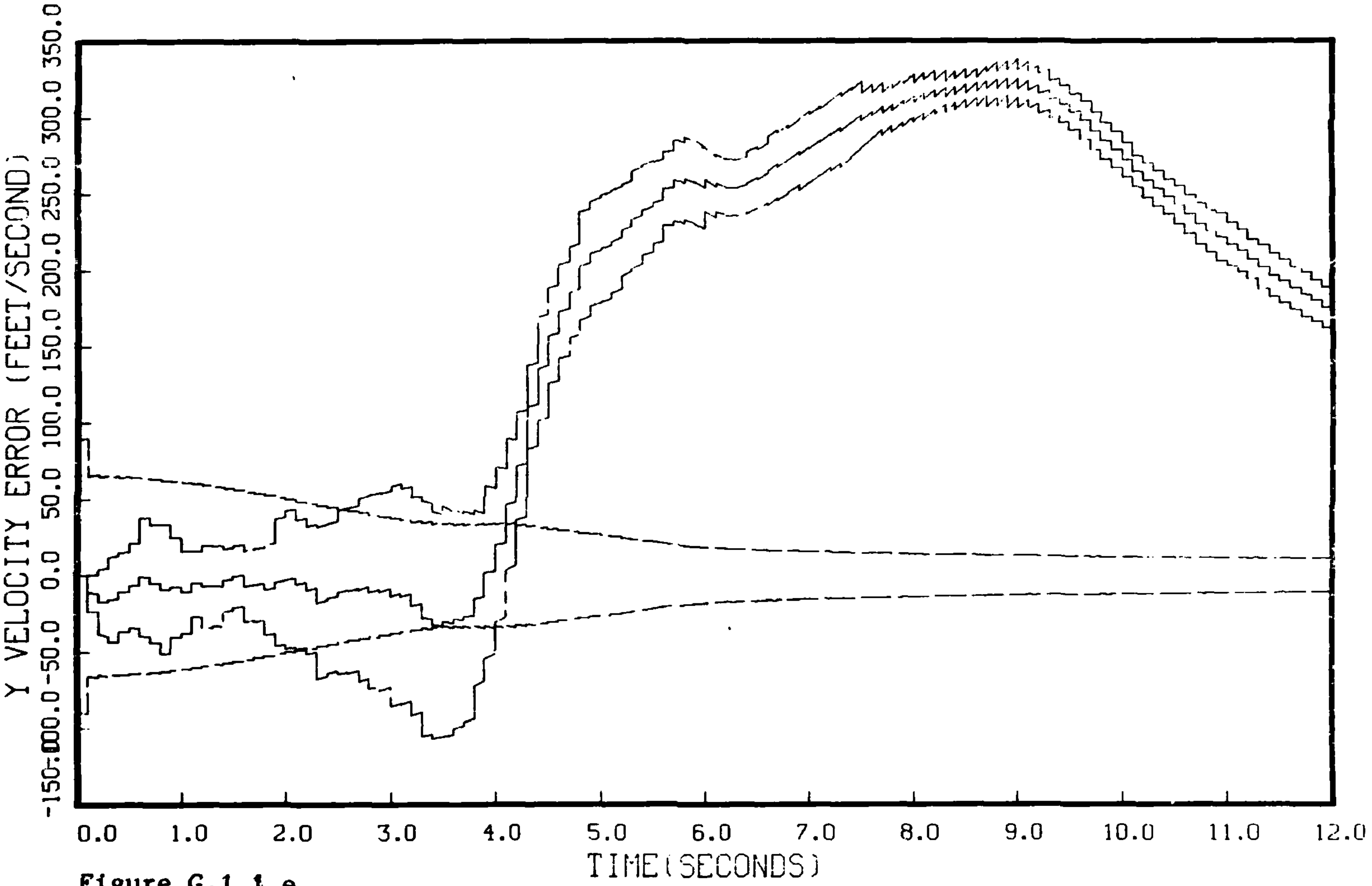


Figure G.1.1.e

STATE 5, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

6 SOFEPL DATE AND TIME - 85/09/30. 19.33.44. SOFE DATE AND TIME - 85/09/30. 18.47.55. WPAFB, DISSPLA, SOFEPL VERSION 2.2

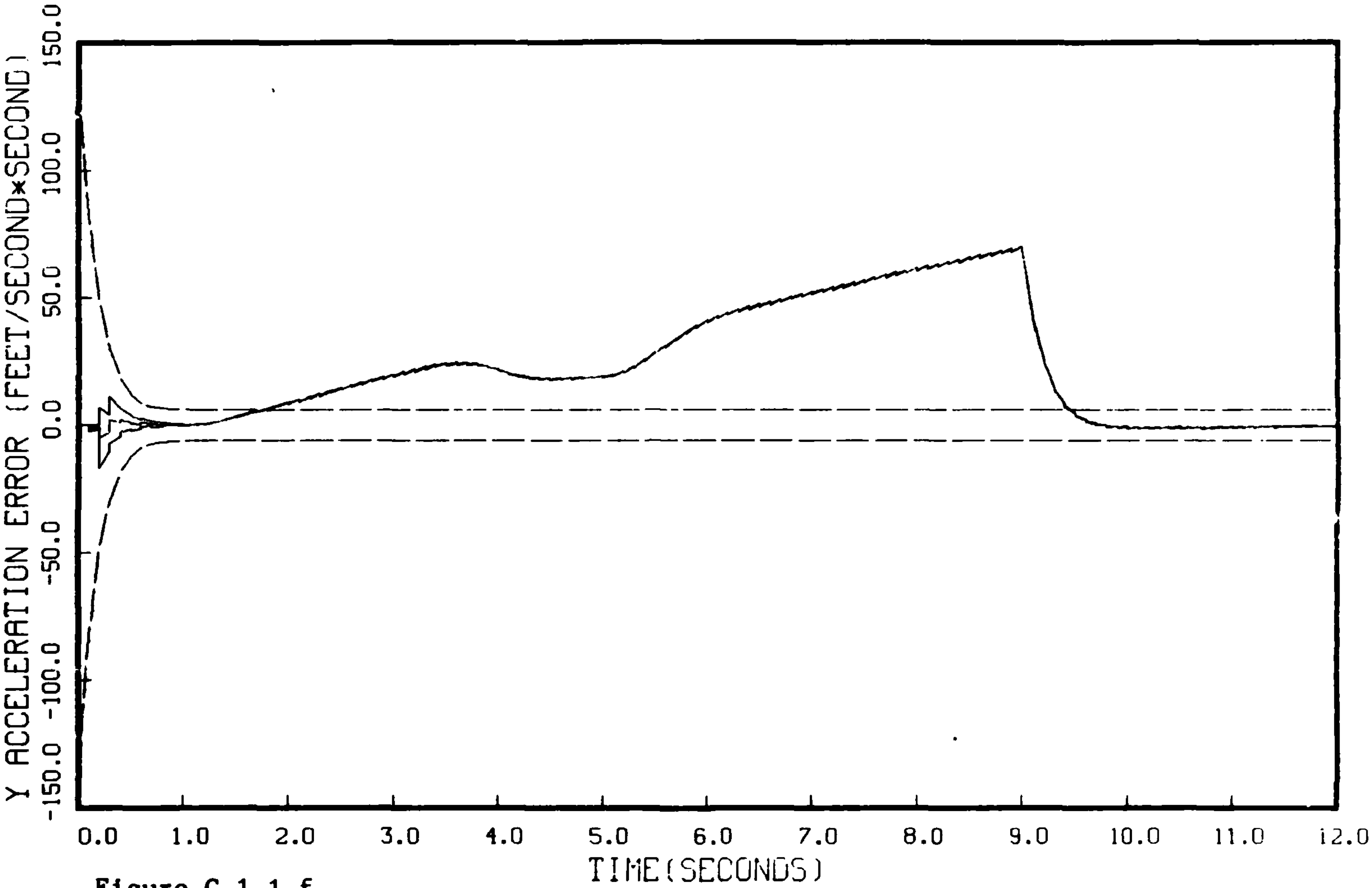


Figure G.1.1.f

STATE 6, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

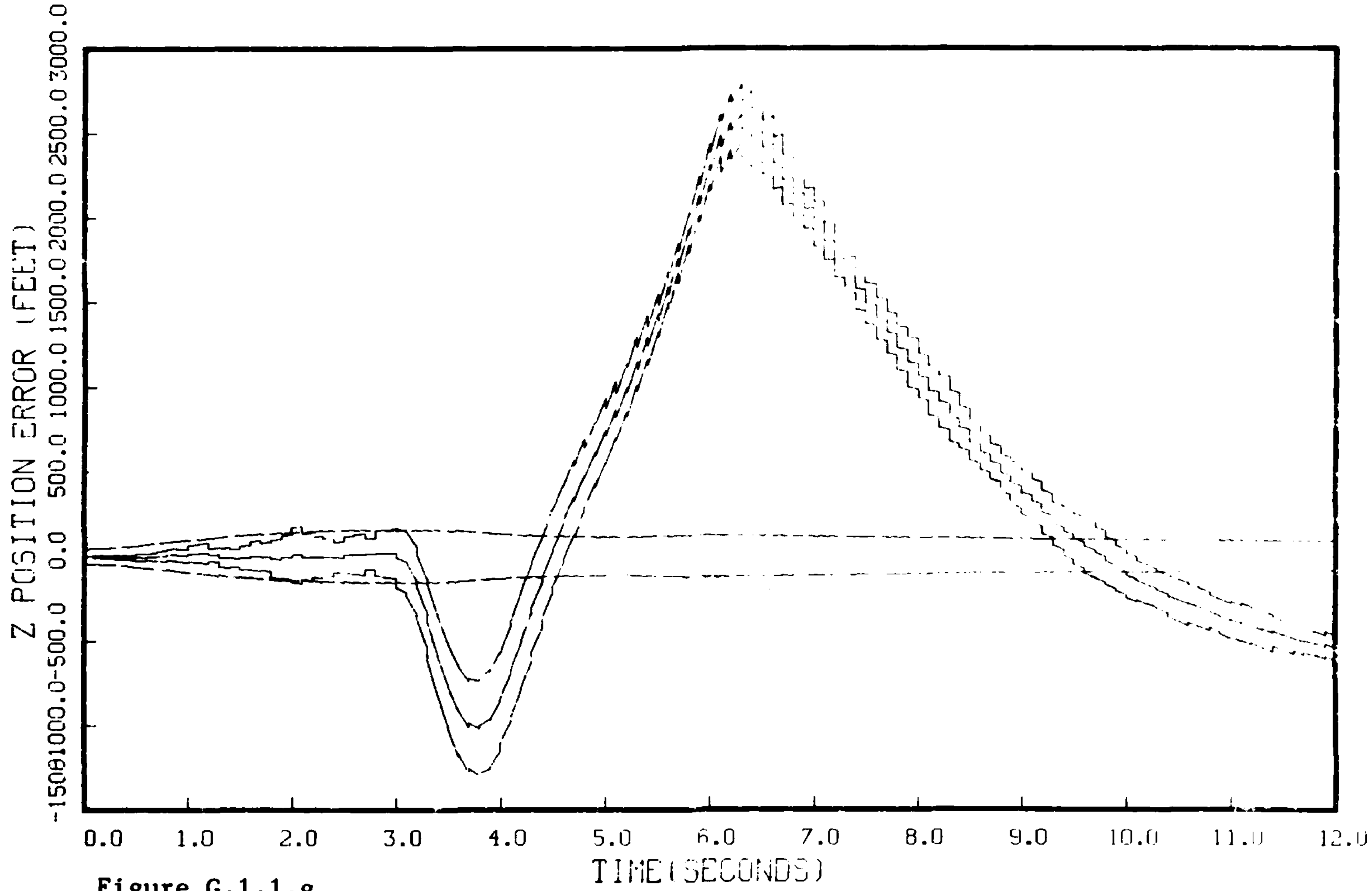


Figure G.1.1.g

STATE 7, Q(1)=Q(2)=Q(3)=375.25, TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

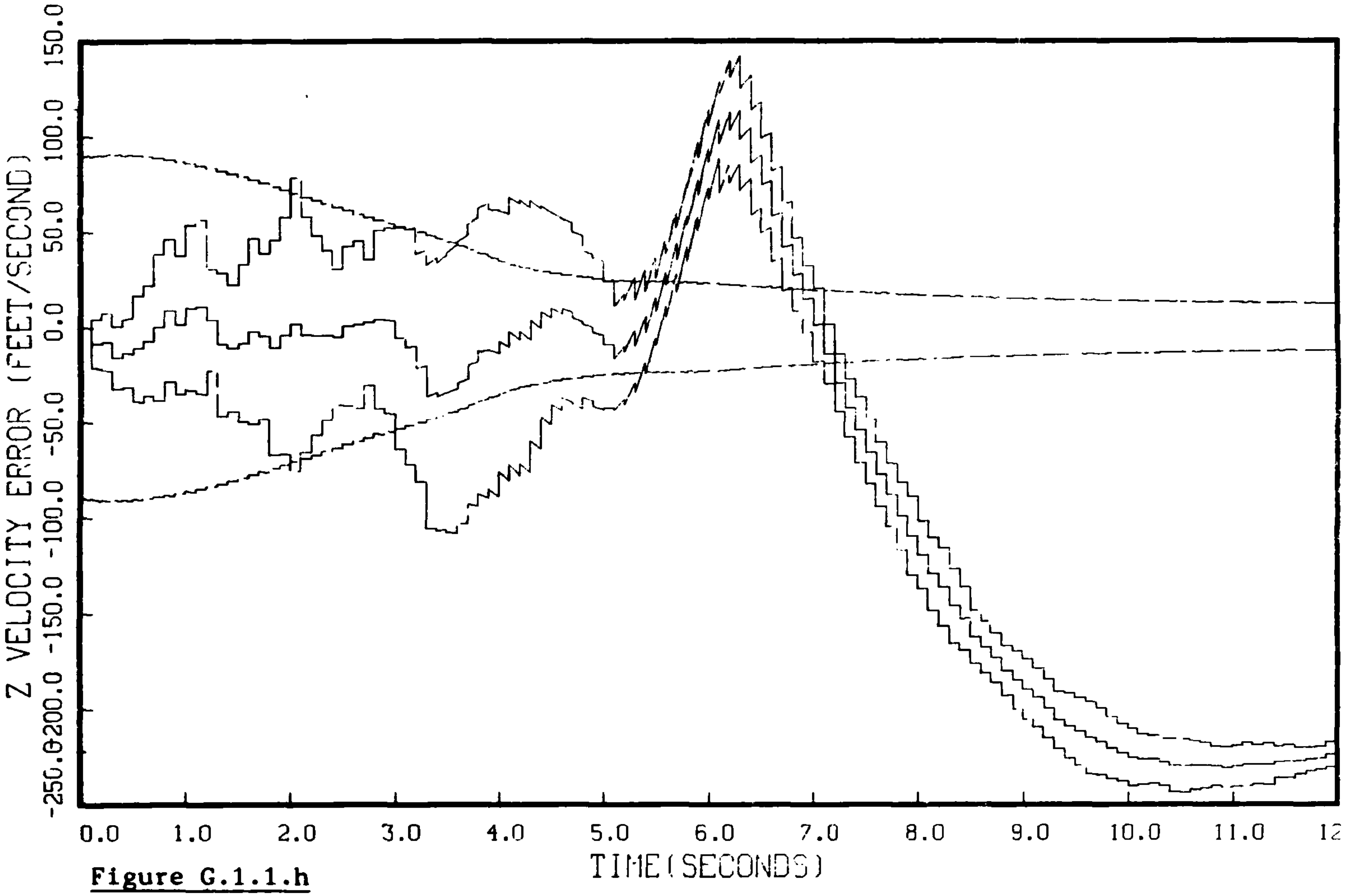


Figure G.1.1.h

STATE 8, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

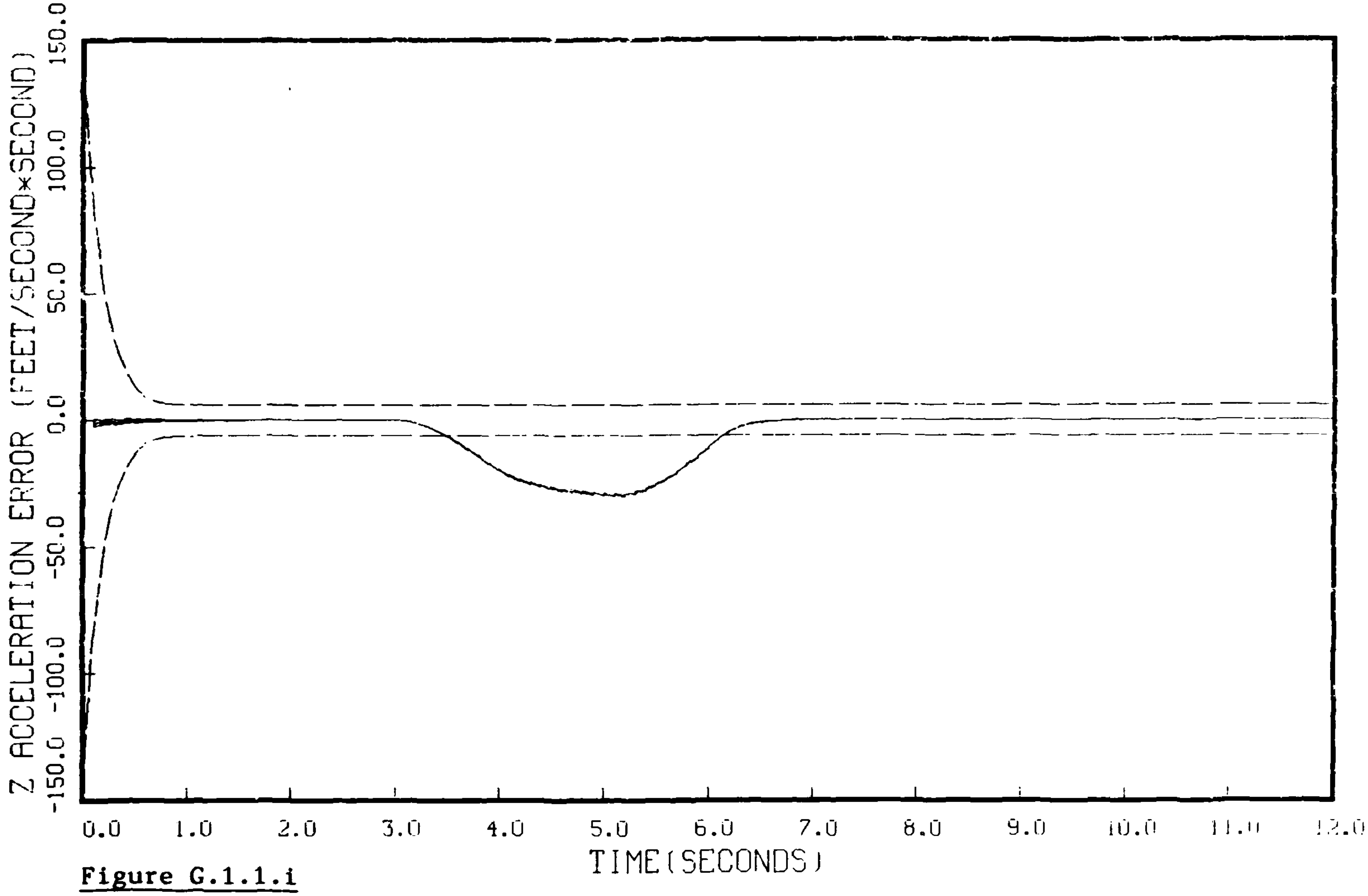


Figure G.1.1.i

STATE 9, Q(1)-Q(2)-Q(3)-373.25, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

PS1 2

SOFEPL DATE AND TIME - 85/09/30. 23.28.18. SOFE DATE AND TIME - 85/09/30. 22.38.08. WPAFB, DIS/FLR, SOFEPL VERSION 2.2

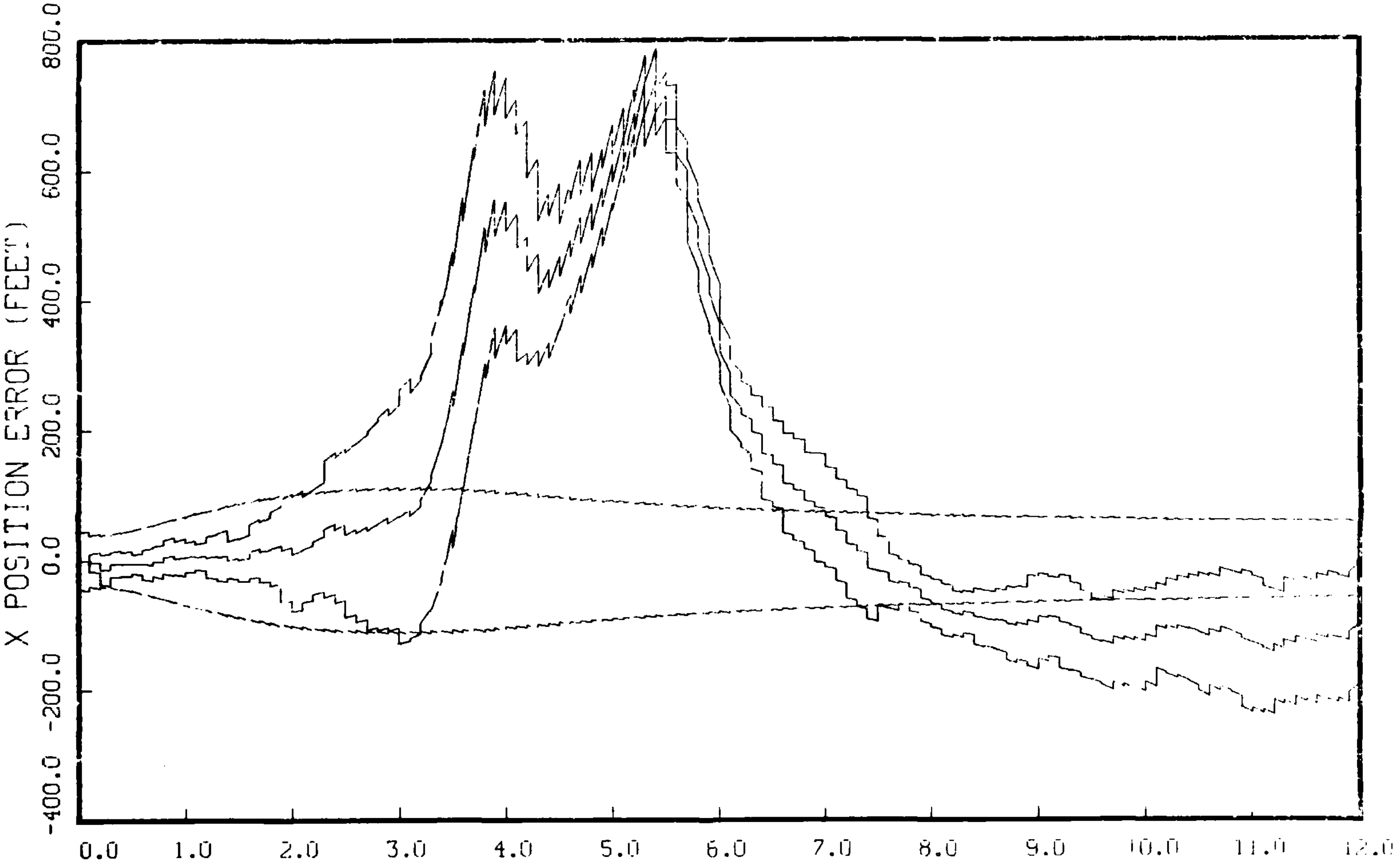


Figure G.1.2.a

STATE 1, Q(1)-Q(2)-Q(3)-3732.5, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

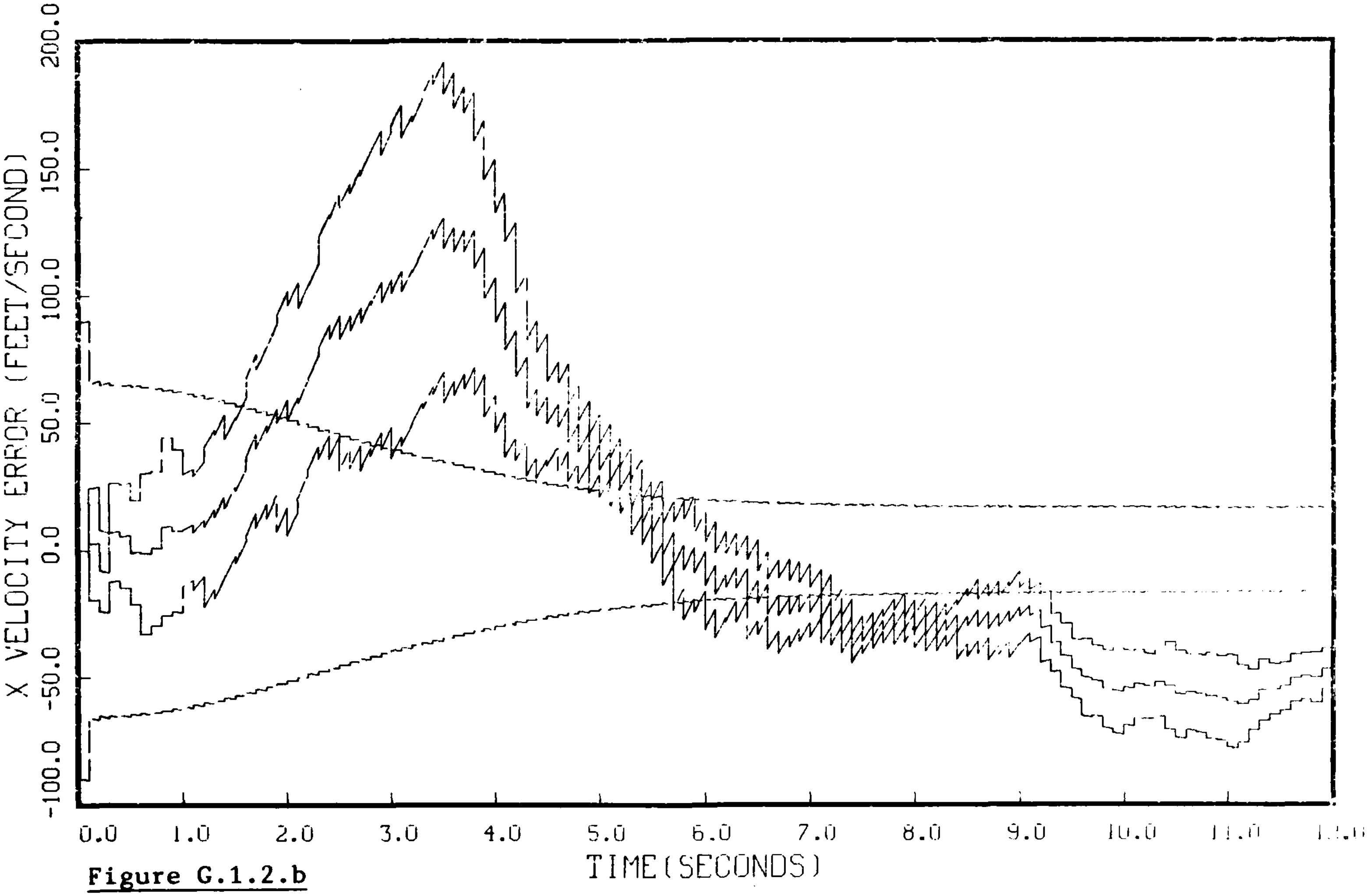


Figure G.1.2.b

STATE 2, Q(1)-Q(2)-Q(3)-3732.5, TAU(1)=.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE=.1, 5 RUNS

3 SUPEPL DATE AND TIME - 85/09/30. 23.28.18. SUPE DATE AND TIME - 85/09/30. 22.38.08. WPAFB, DISSPLA, SUPEPL VERSION 2.2

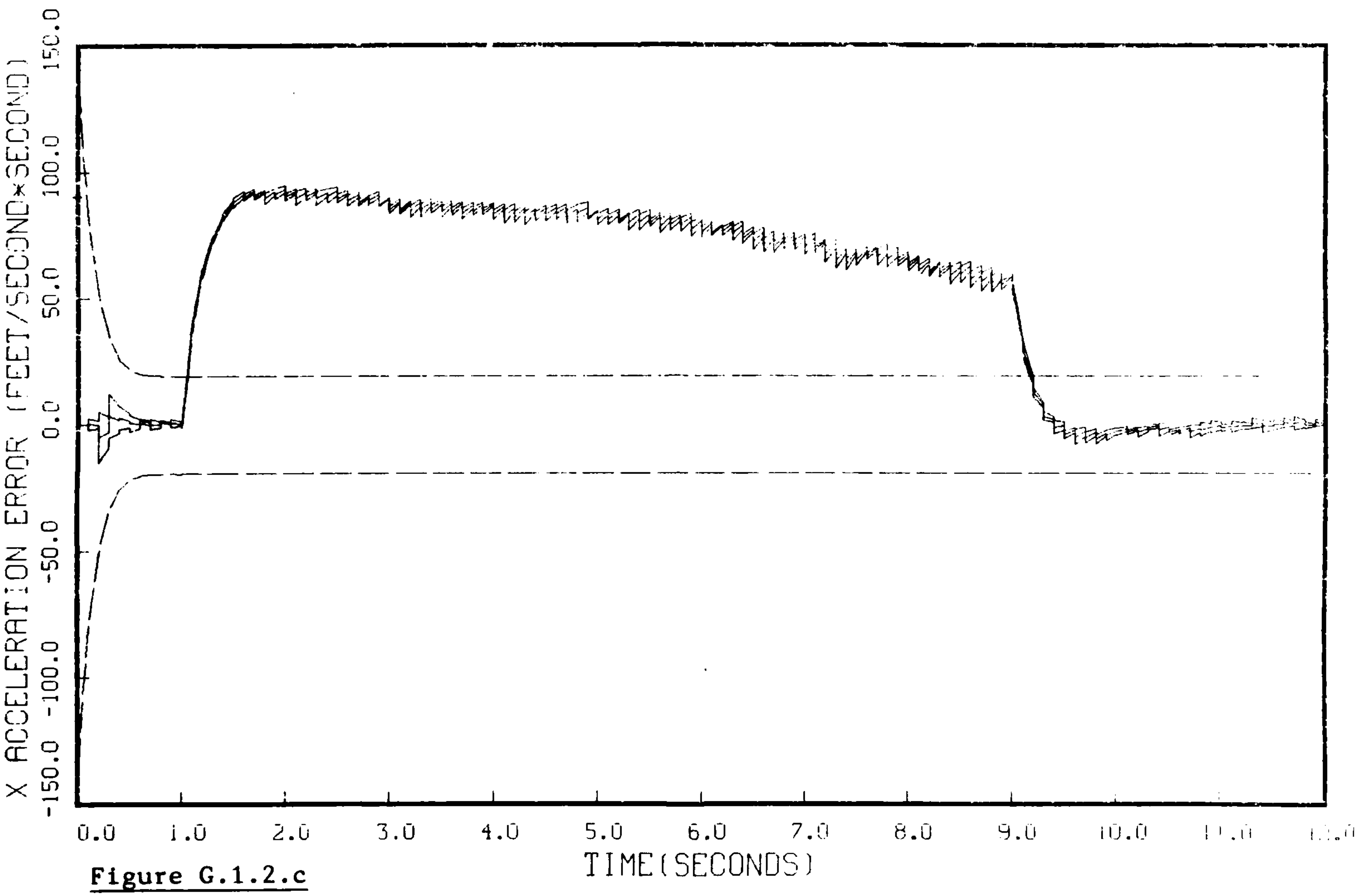


Figure G.1.2.c

STATE 3, Q(1)=Q(2)=Q(3)=3732.5, TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

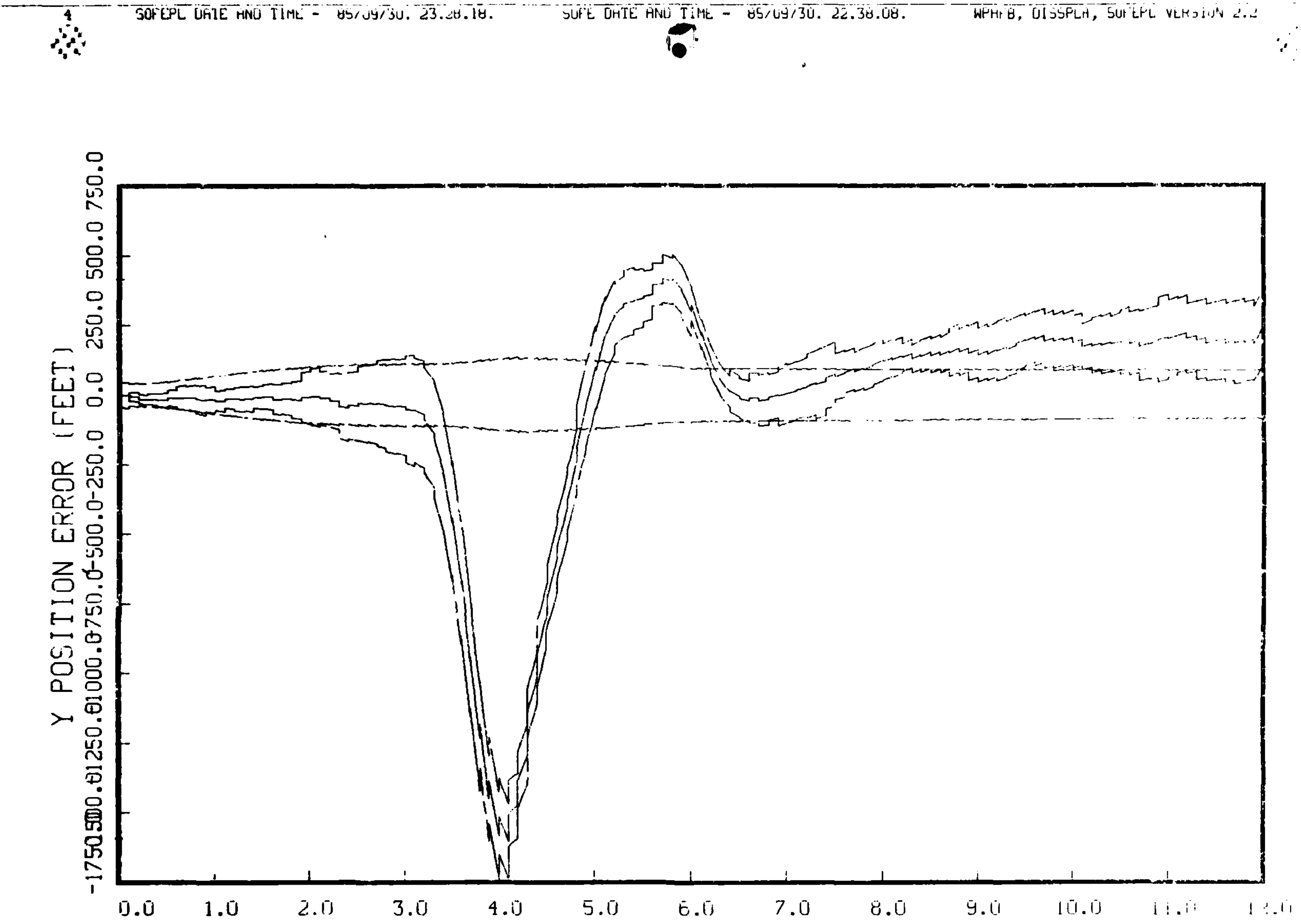


Figure G.1.2.d

STATE 4, Q(1)-Q(2)-Q(3)-3732.5, TAU(1)=.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

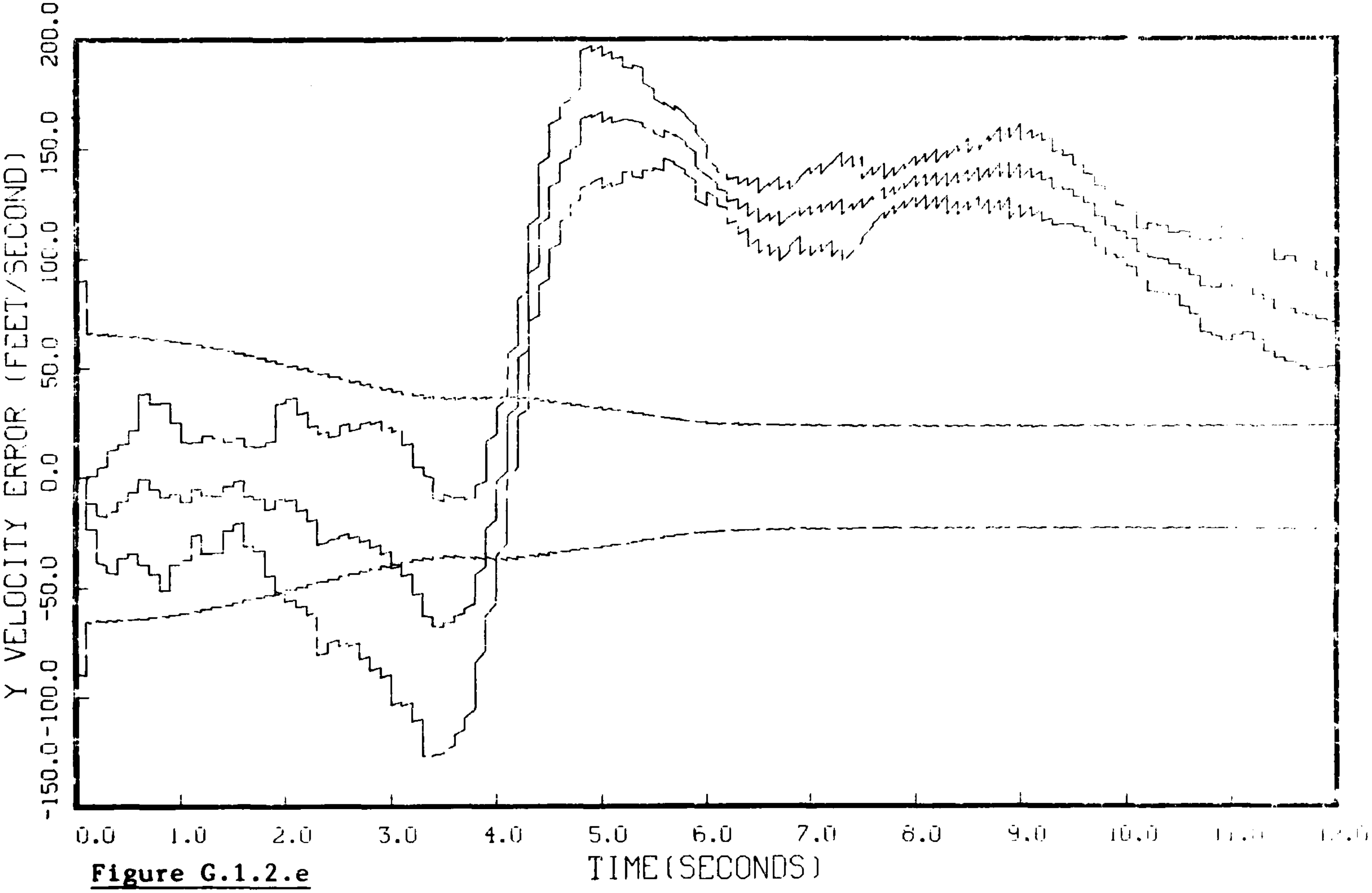


Figure G.1.2.e

STATE 5, Q(1)=Q(2)=Q(3)=3732.5, TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APQ 120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

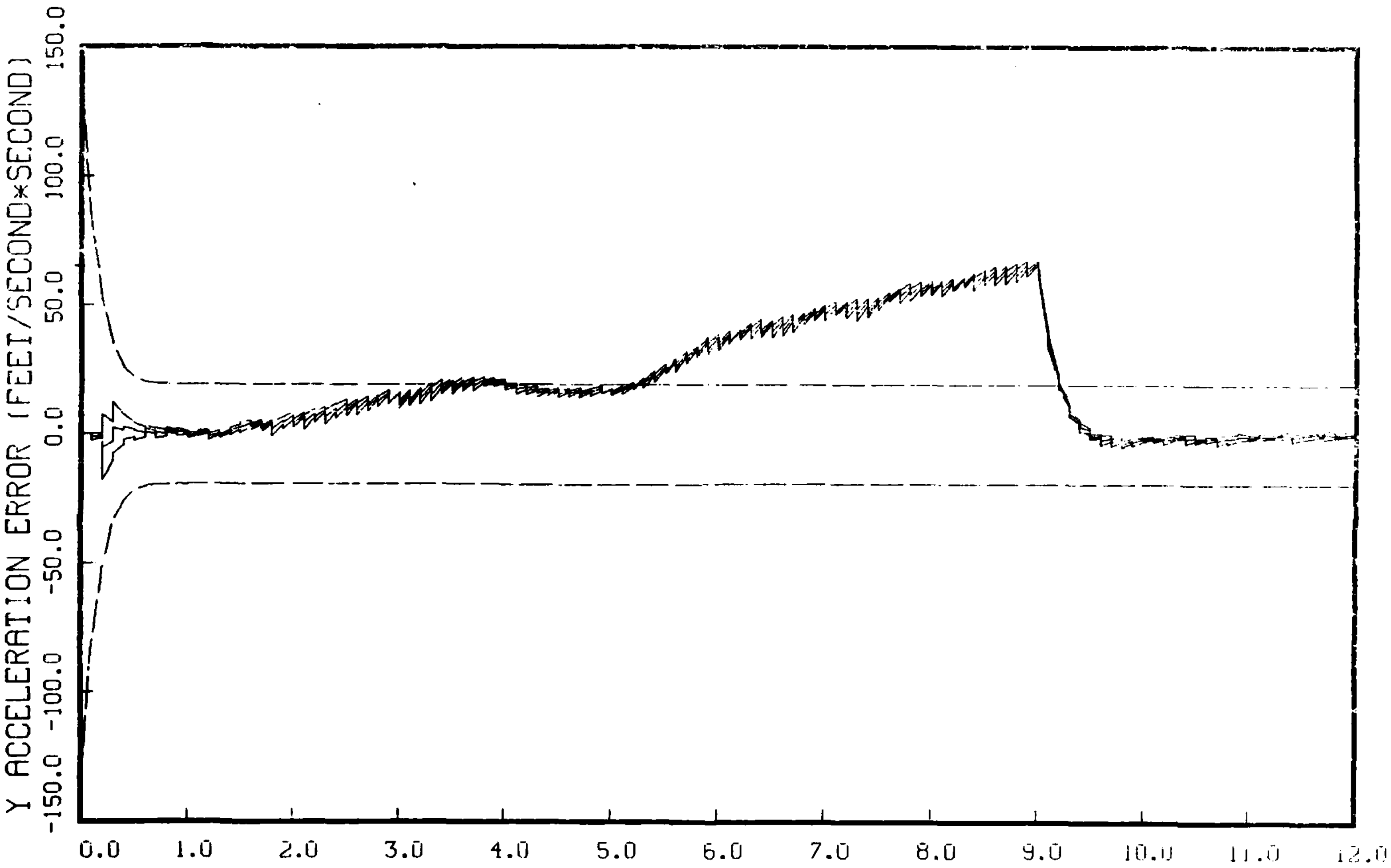


Figure G.1.2.f

STATE 6, Q(1)-Q(2)-Q(3)-3732.5, TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

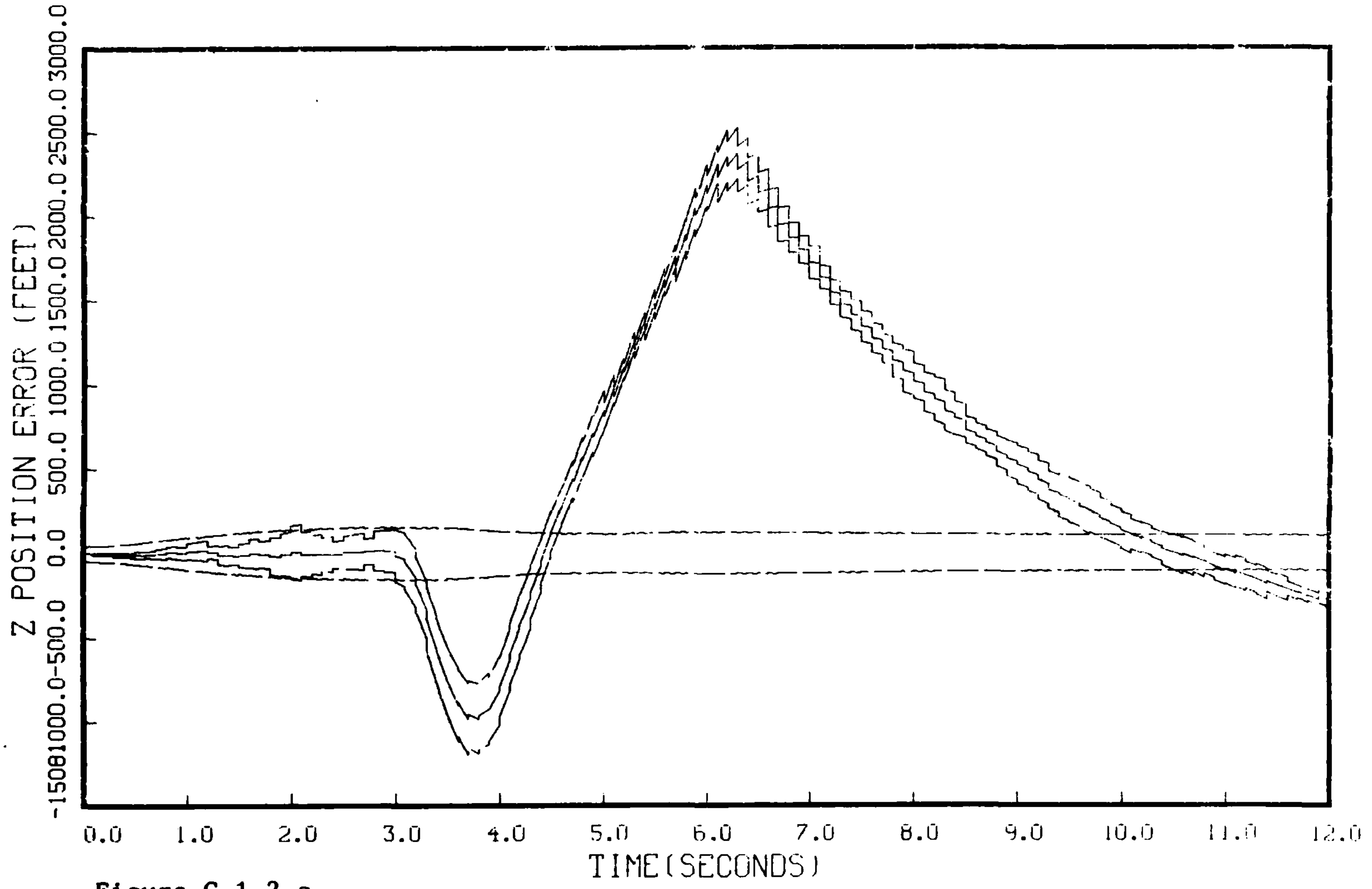


Figure G.1.2.g
STATE 7, Q(1)-Q(2)-Q(3)-3732.5, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

SUFEPL DATE AND TIME - 85/09/30. 23.28.13.   SUFE DATE AND TIME - 85/09/30. 22.38.08.   WPAFB, UTSSPLA, SUFEPL VERSION 2.2

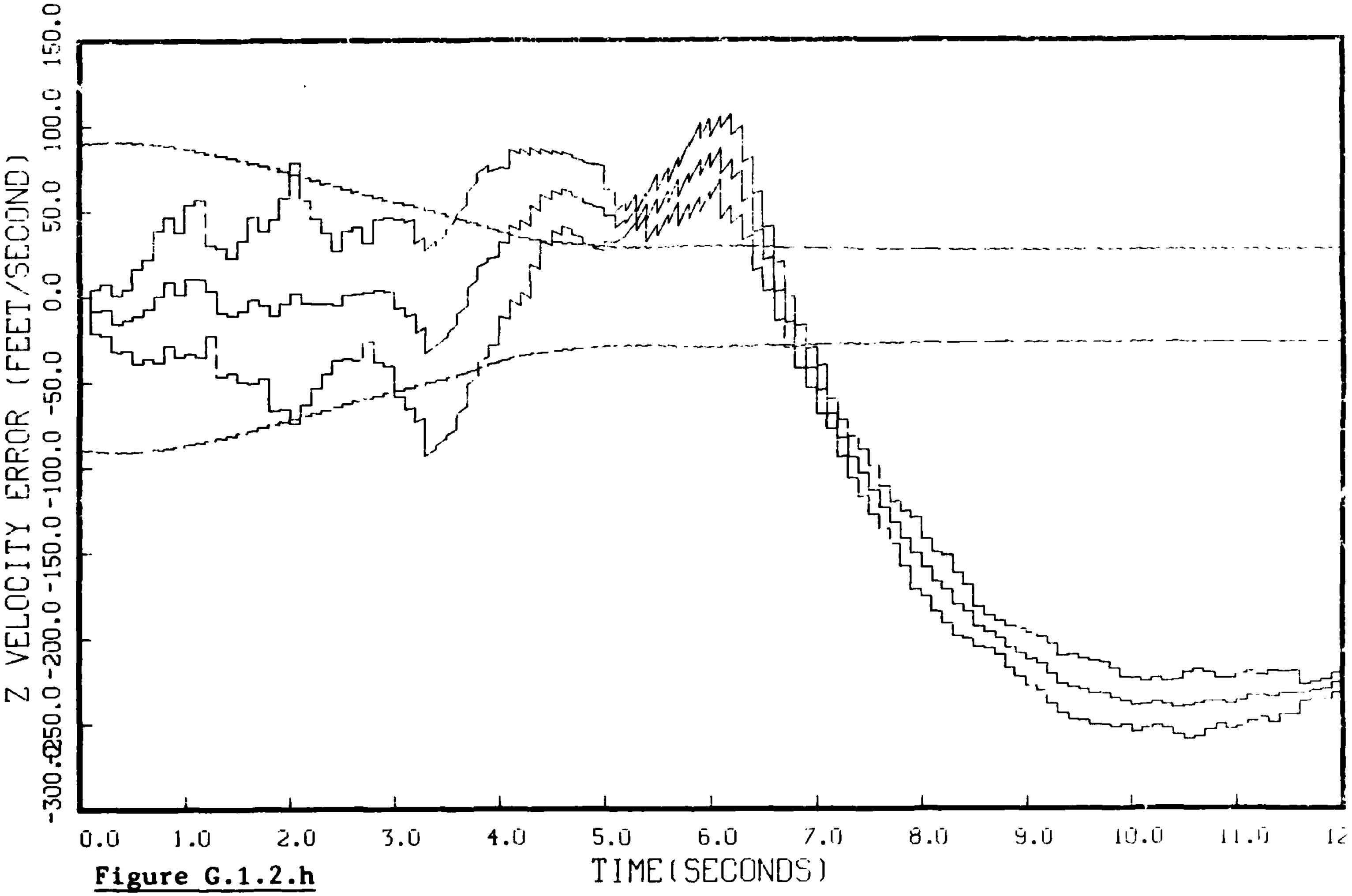


Figure G.1.2.h

STATE 8, Q(1)-Q(2)-Q(3)-3732.5, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
AFO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

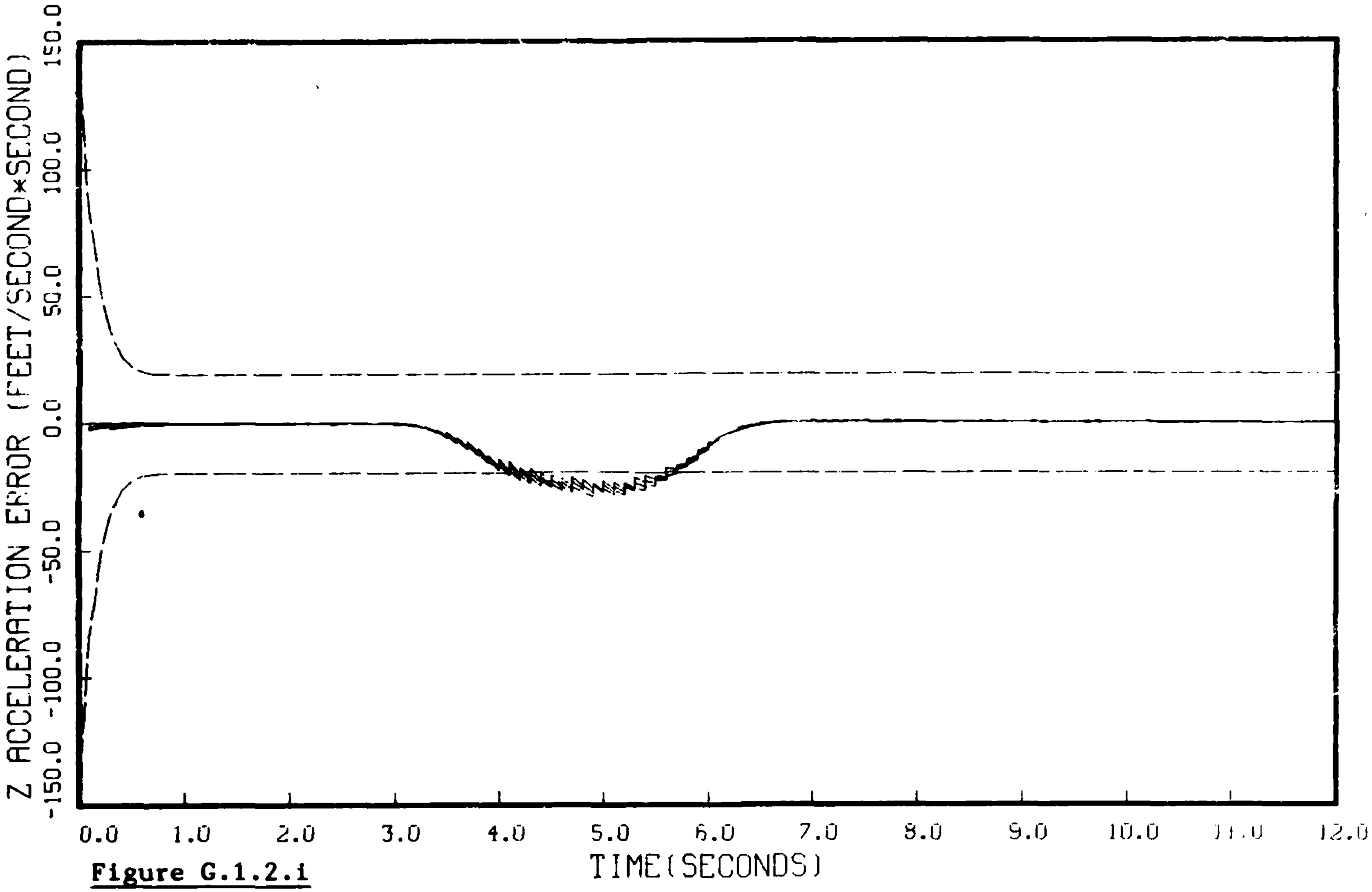


Figure G.1.2.i

STATE 9, Q(1)-Q(2)-Q(3)-3732.5, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

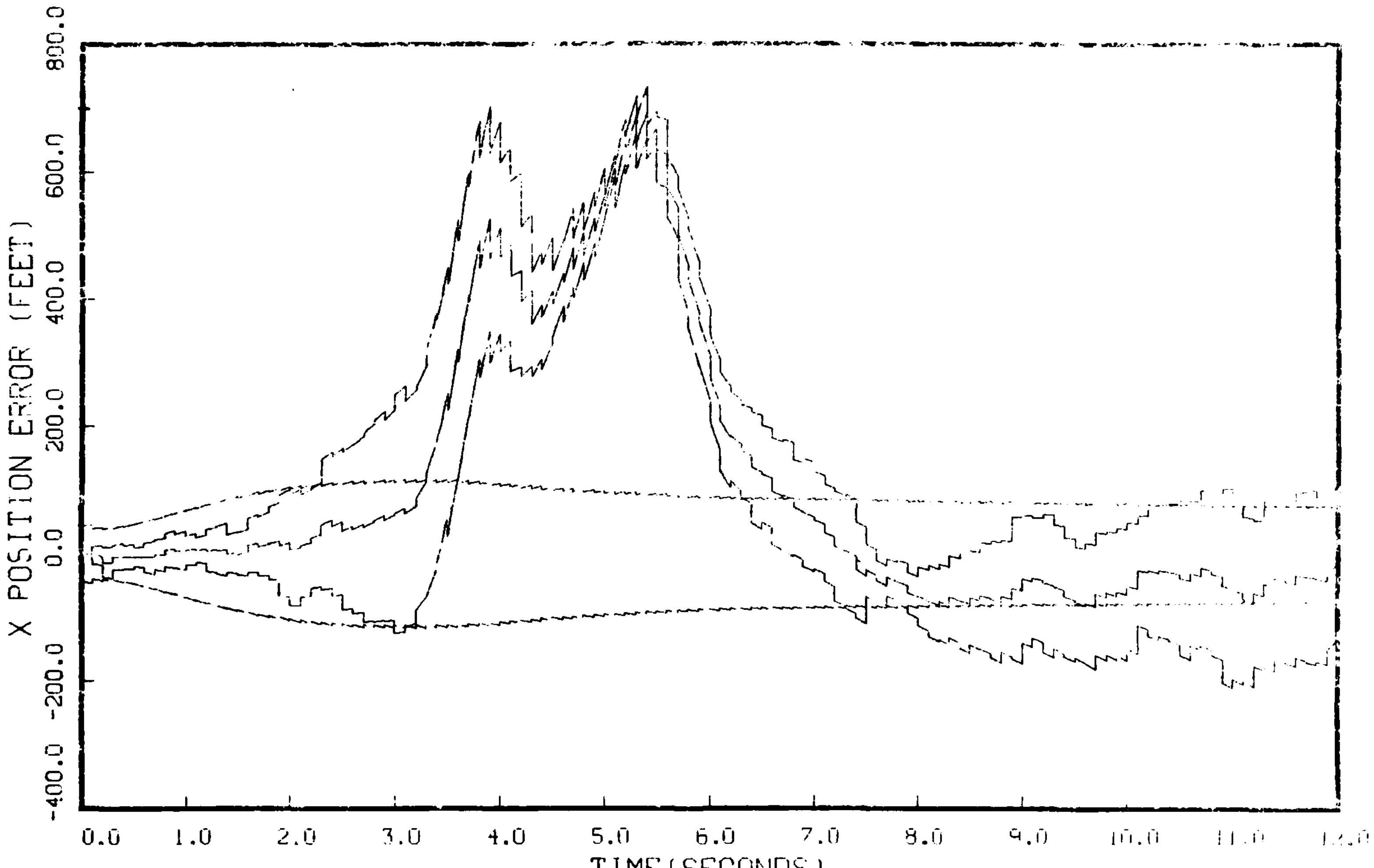


Figure G.1.3.a

STATE 1, Q(1)=Q(2)=Q(3)=3732 5, TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

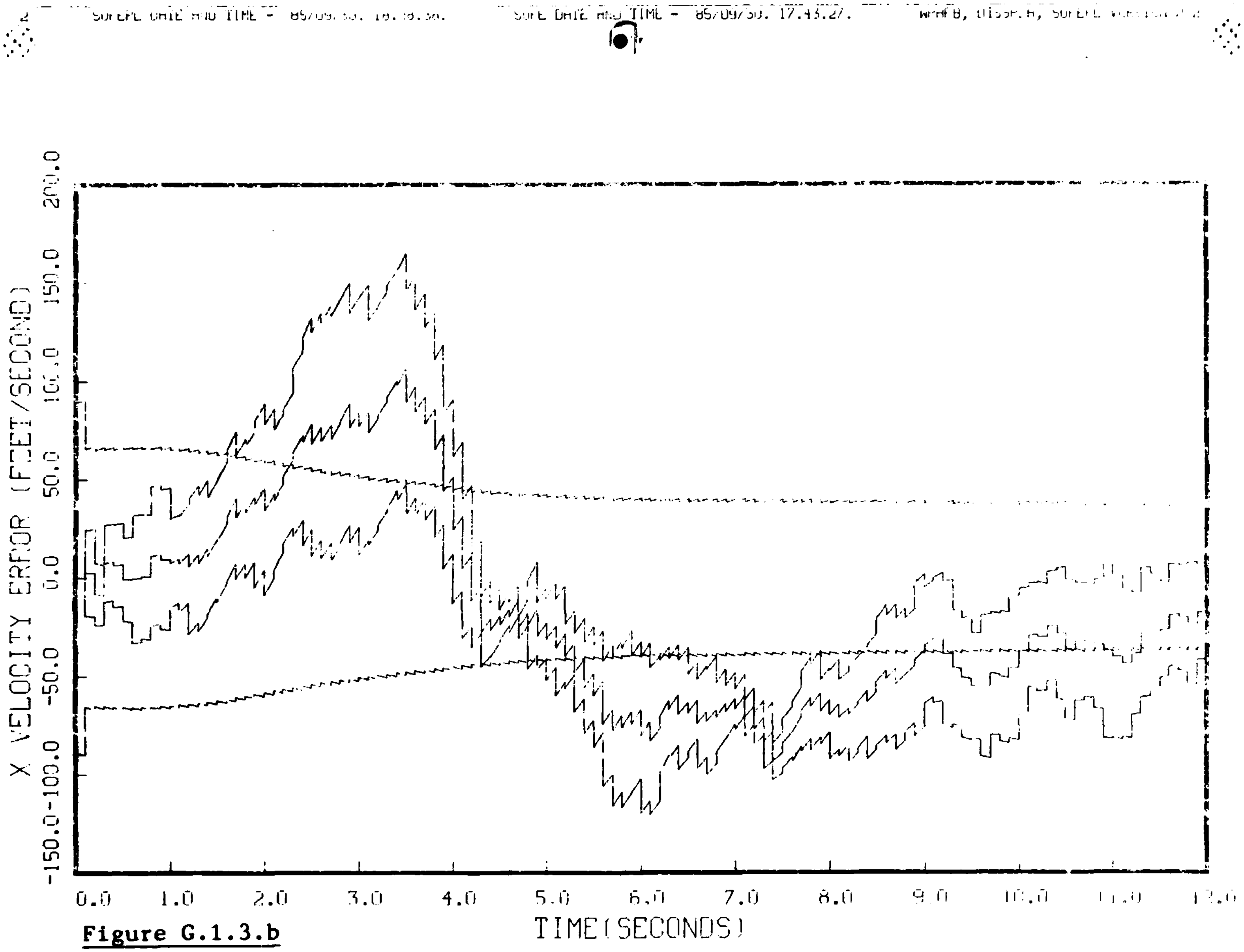


Figure G.1.3.b

STATE 2, O(1)-O(2)-O(3)-3732 5, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
ARS-120, BEAM ATTACK, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

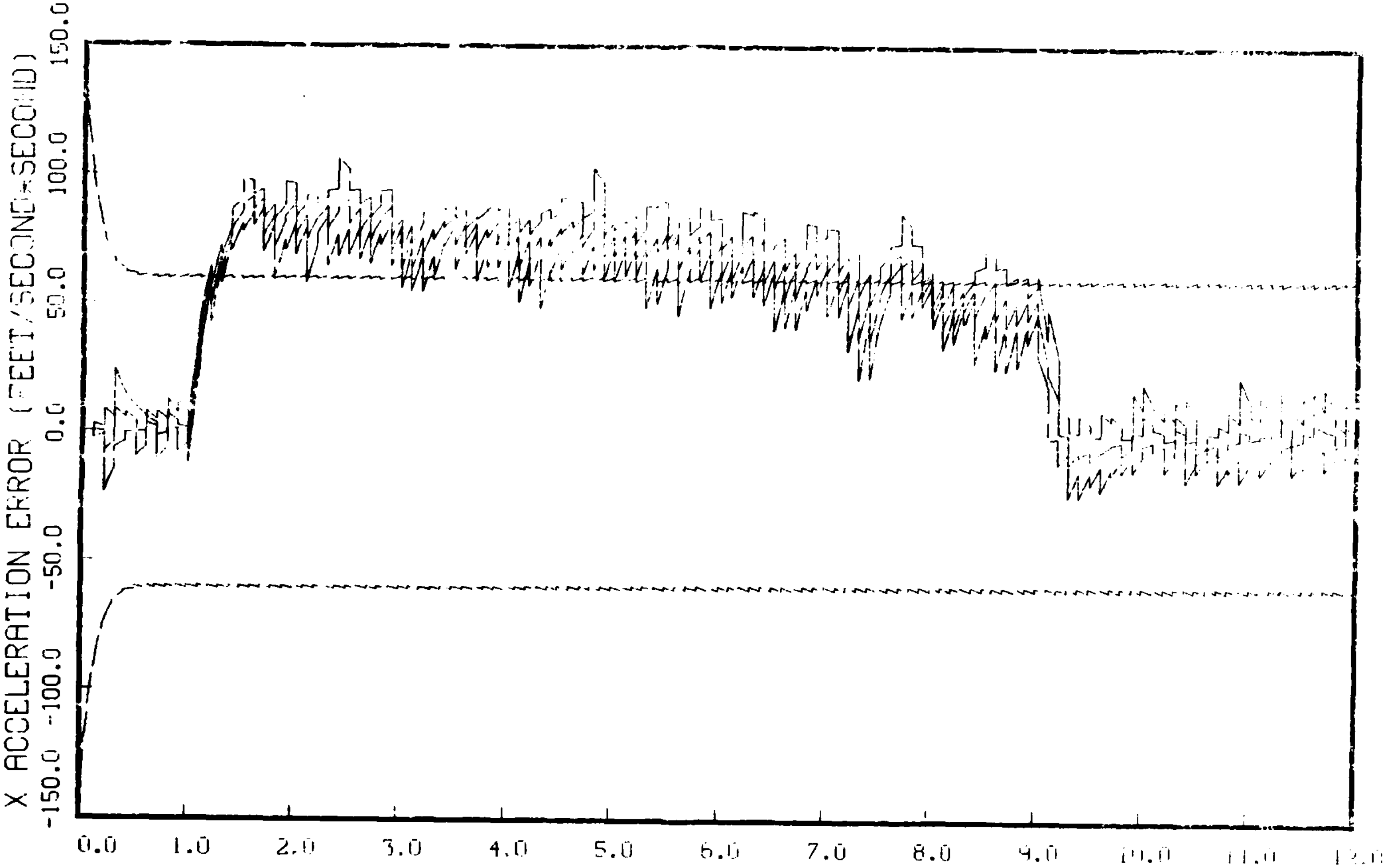


Figure G.1.3.c

STATE 3, Q(1)-Q(2)-Q(3)-3732 5., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

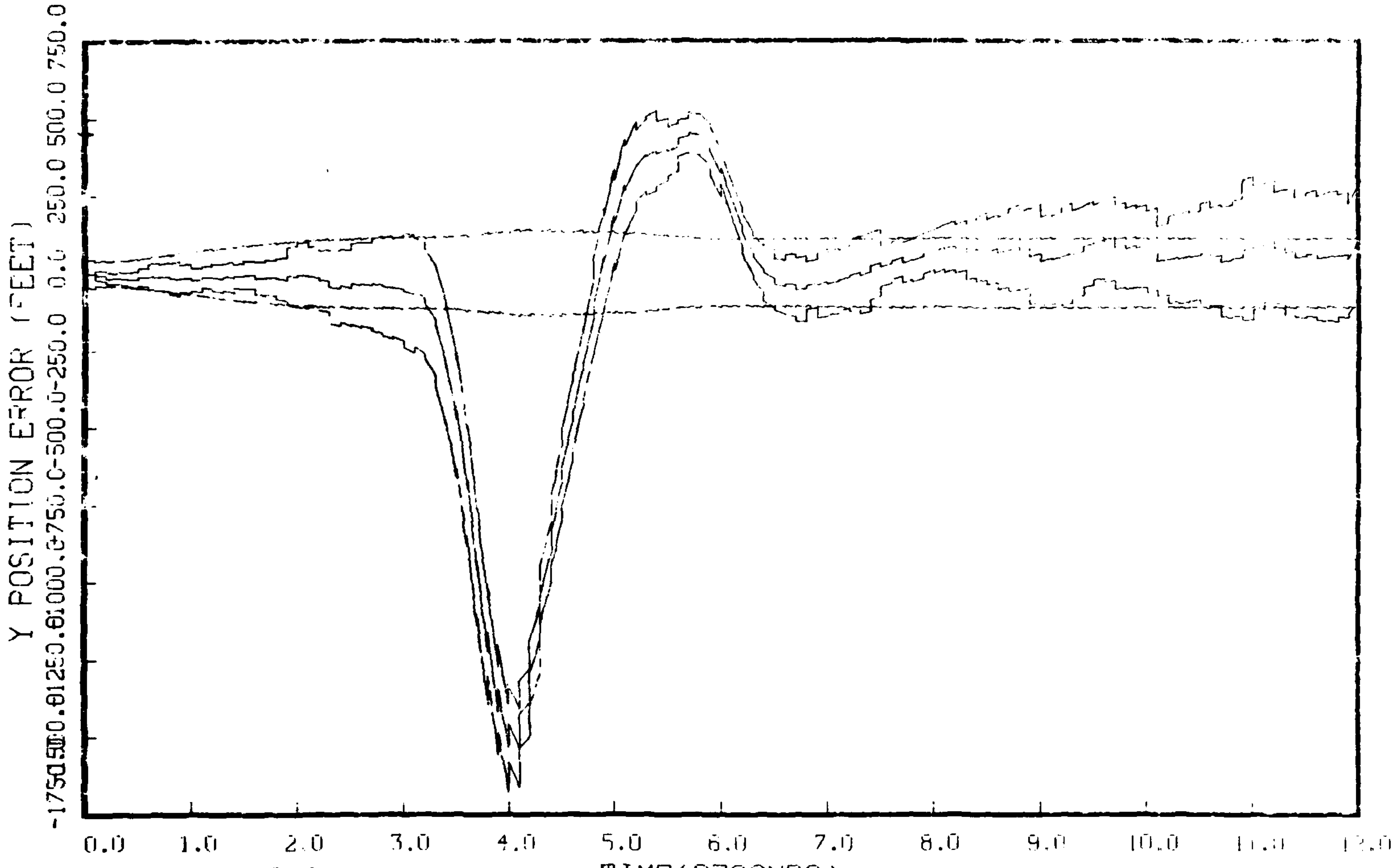


Figure G.1.3.d

STATE 4, Q(1)-Q(2)-Q(3)-3732 5, TAU(1)-.2,TAU(2-3)=.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

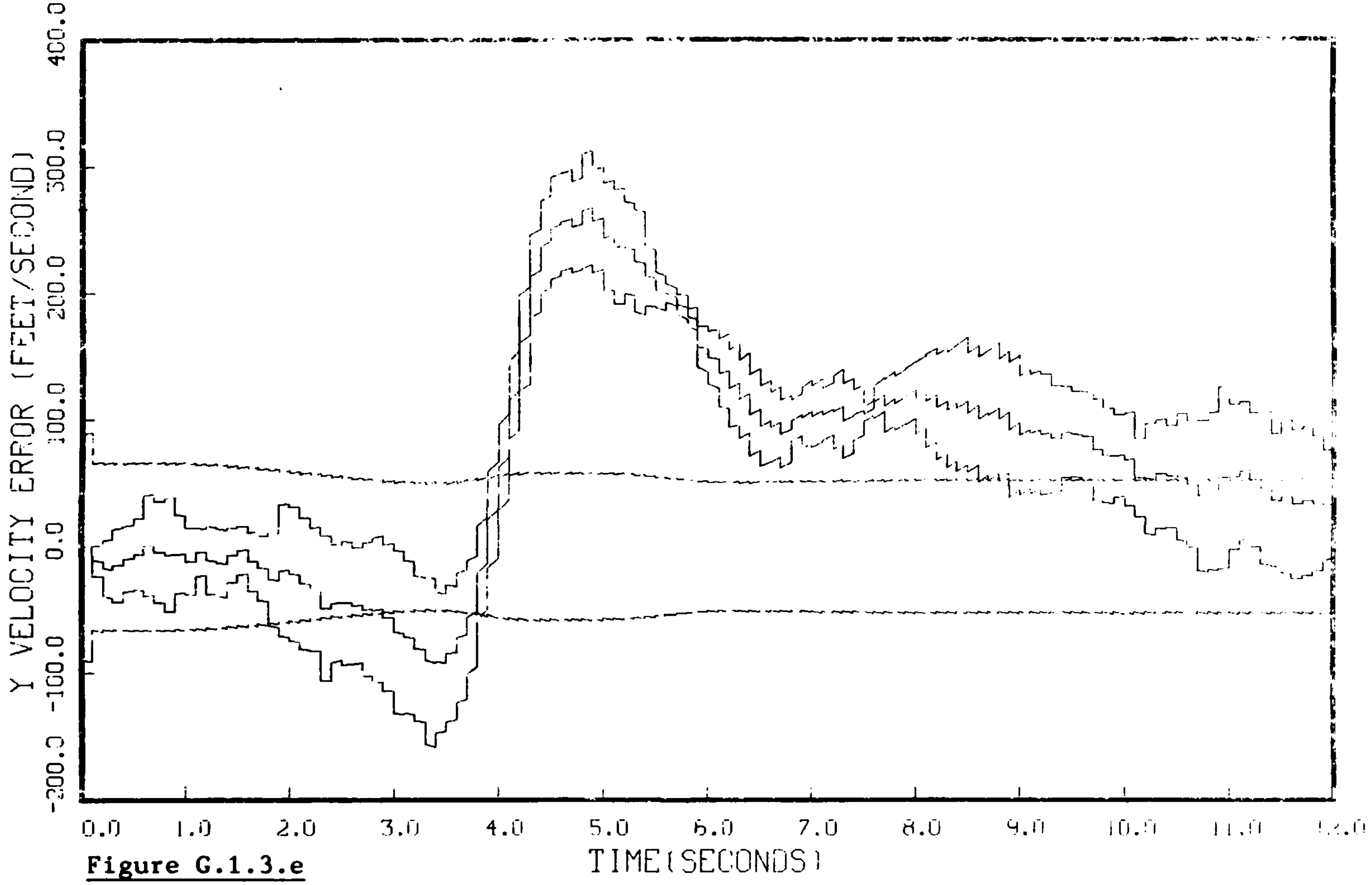


Figure G.1.3.e

STATE 5, Q(1)-Q(2)-Q(3)-3732 5., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

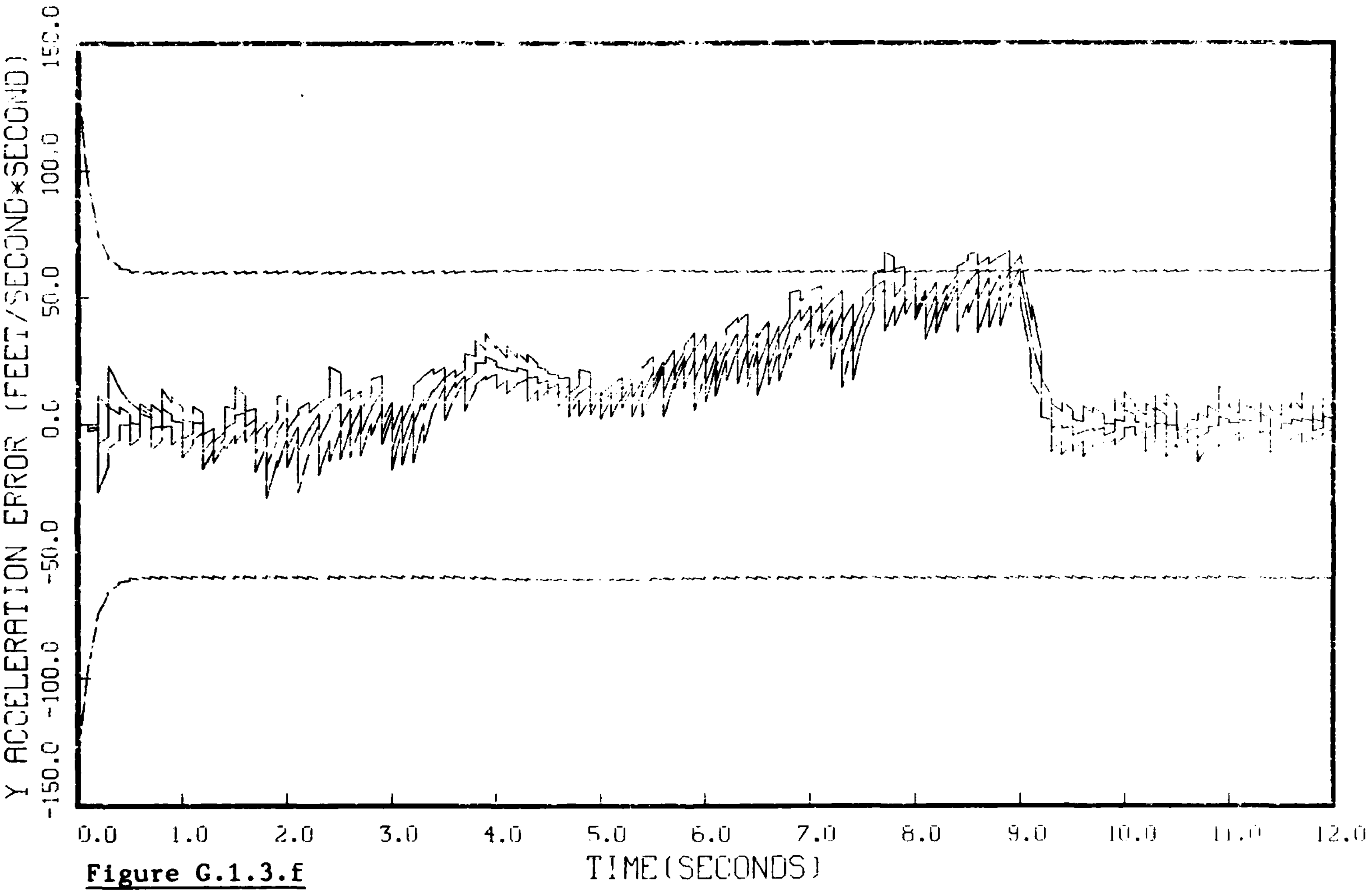


Figure G.1.3.f

STATE 6, Q(1)-Q(2)-Q(3)-3732 5., TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APO 120, BEAM ATTACK, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

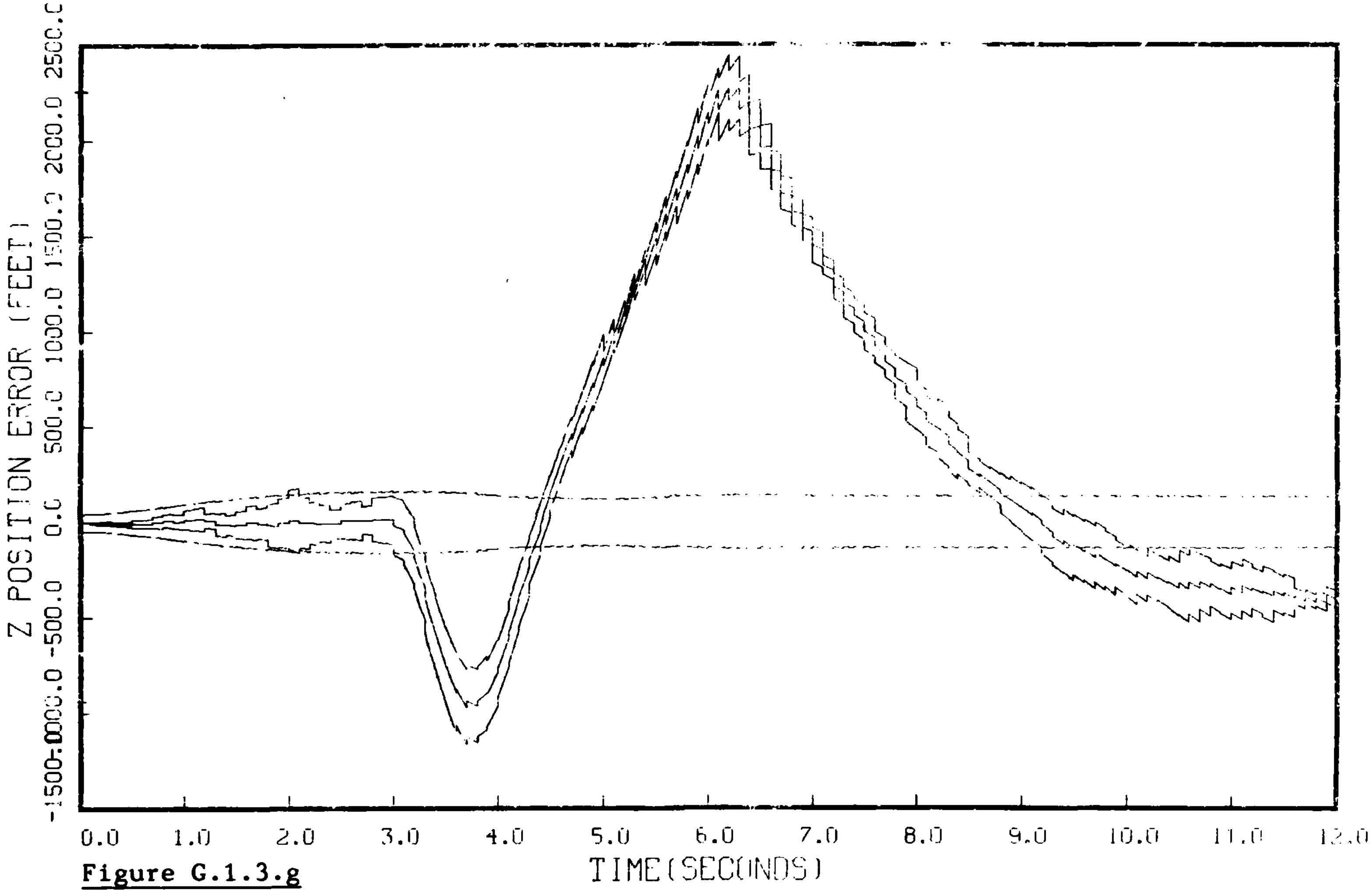


Figure G.1.3.g

STATE 7, Q(1)-Q(2)-Q(3)-3732 5., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

8 SOFEPL DATE AND TIME - 85/09/30. 18.28.38. SOFE DATE AND TIME - 85/09/30. 17.43.27. WPAFB, DISSPLA, SOFEPL VERSION 2.2

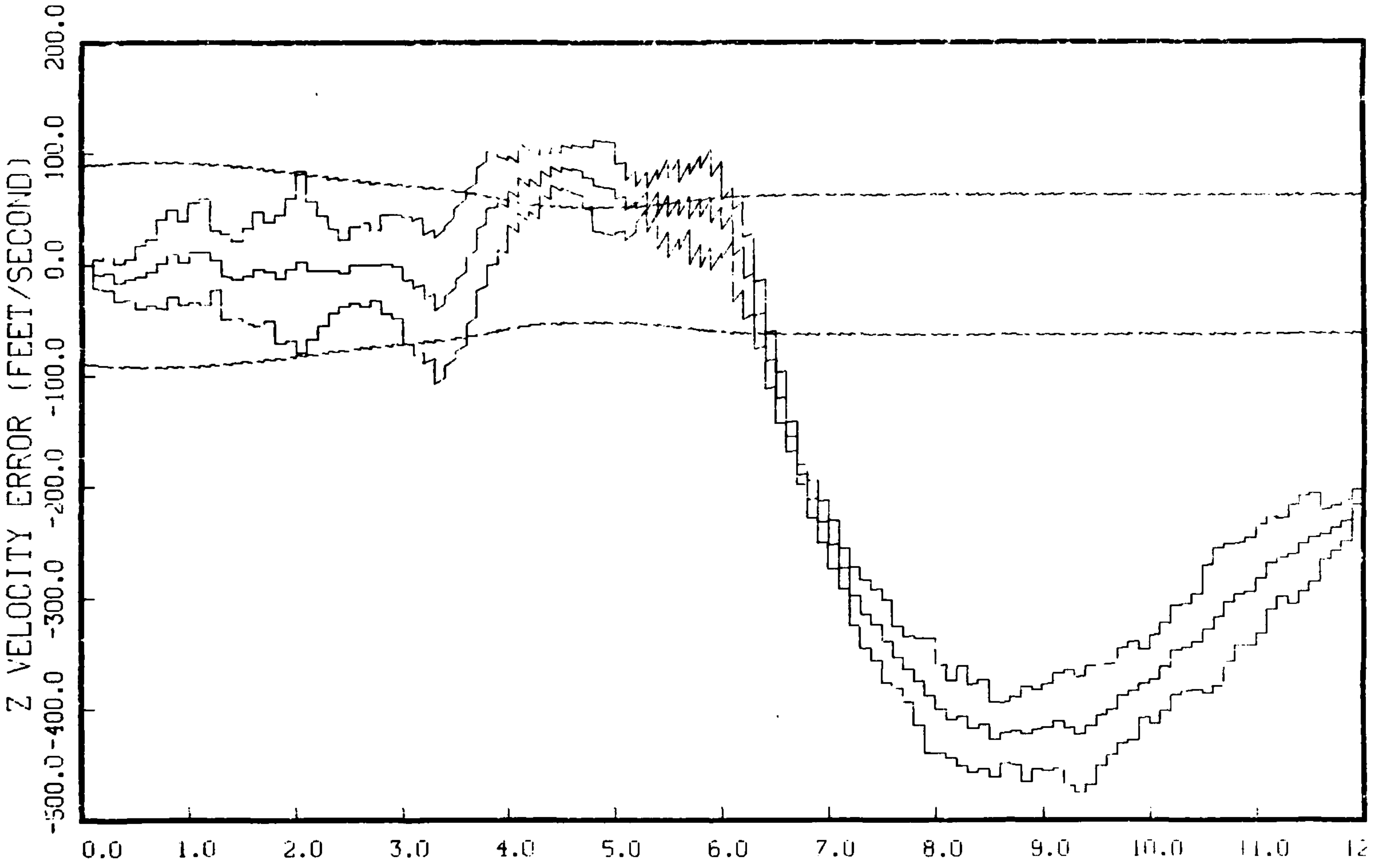


Figure G.1.3.h

STATE 8, Q(1)-Q(2)-Q(3)-3732 5, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

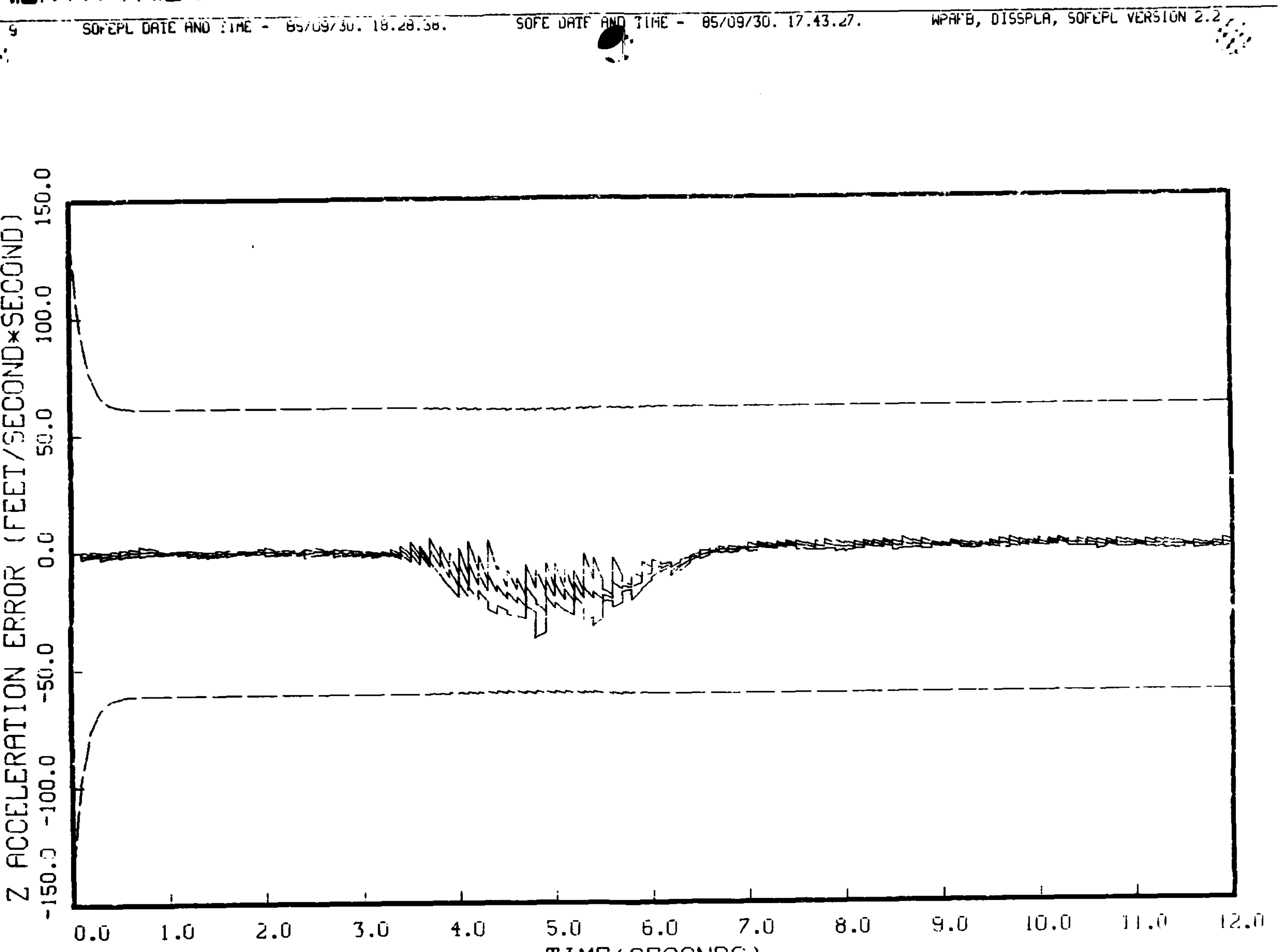


Figure G.1.3.i

STATE 9, Q(1)-Q(2)-Q(3)-3732 5, TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

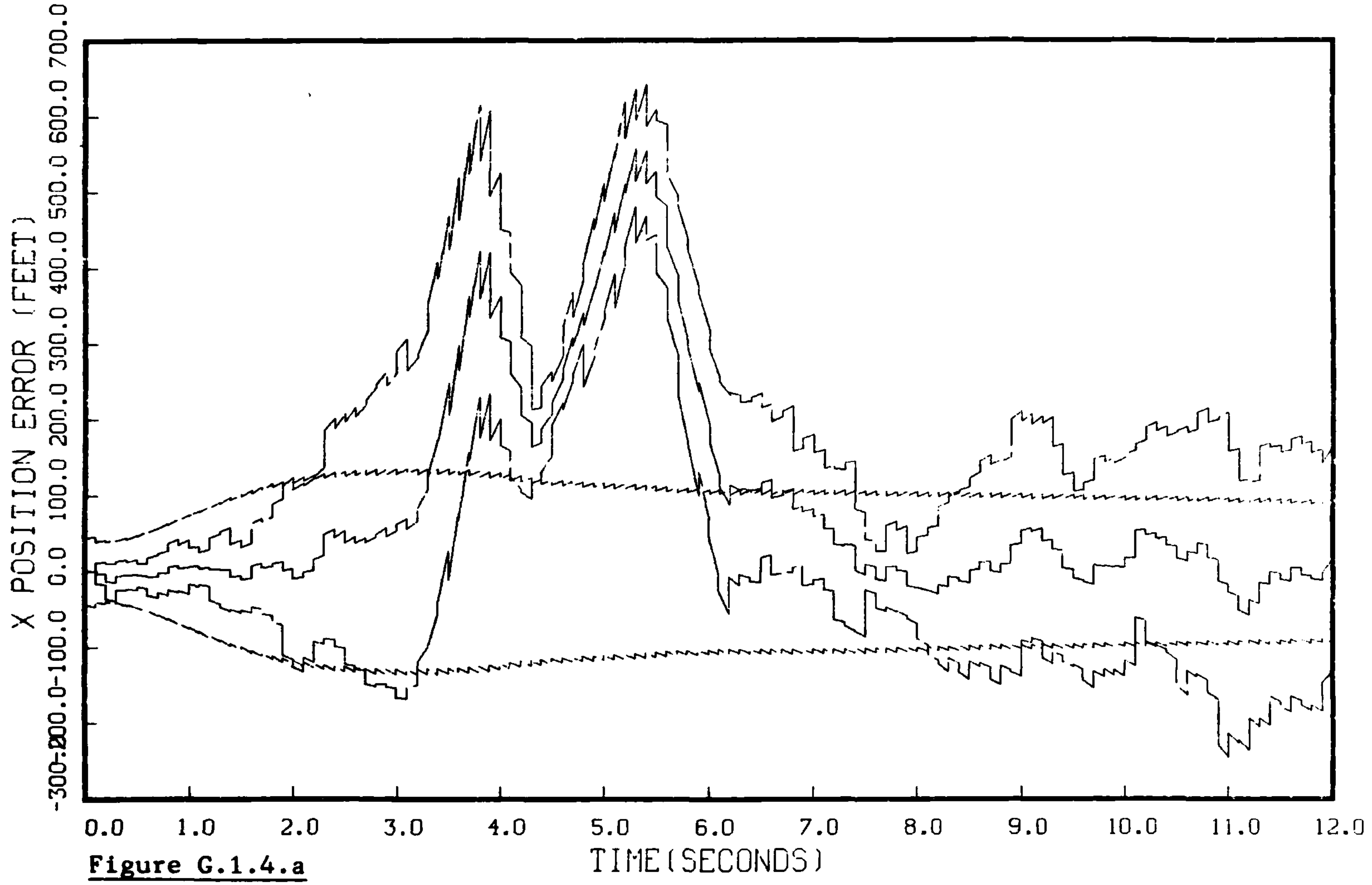


Figure G.1.4.a

STATE 1, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

2 SOFEPL DATE AND TIME - 85/10/01. 17.08.31. SOFE DATE AND TIME - 85/10/01. 16.33.53. WPAFB, DISSPLA, SOFEPL VERSION 2.2

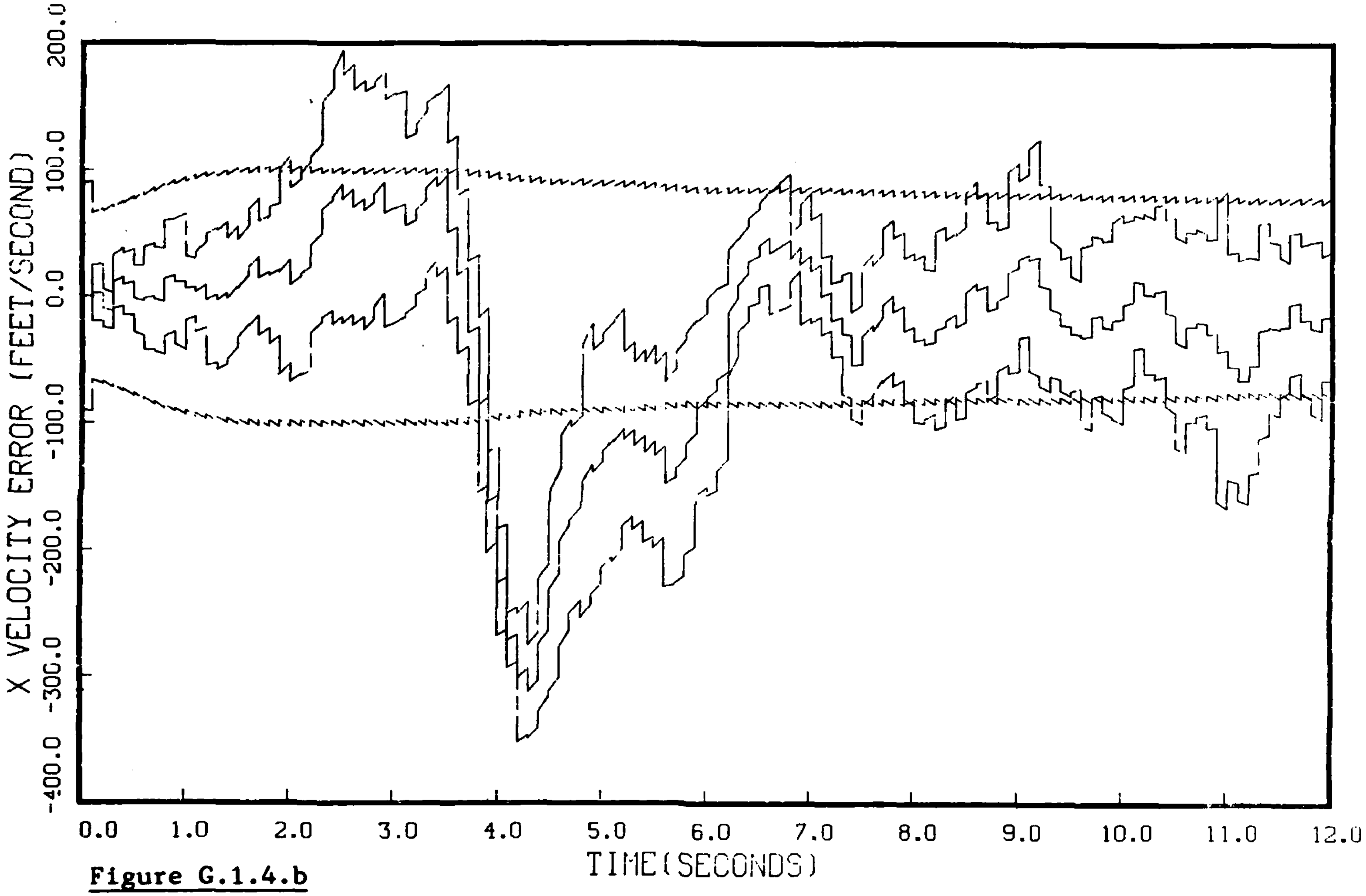


Figure G.1.4.b

STATE 2, Q(1)-Q(2)-Q(3)-373250., TAU(1)=.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

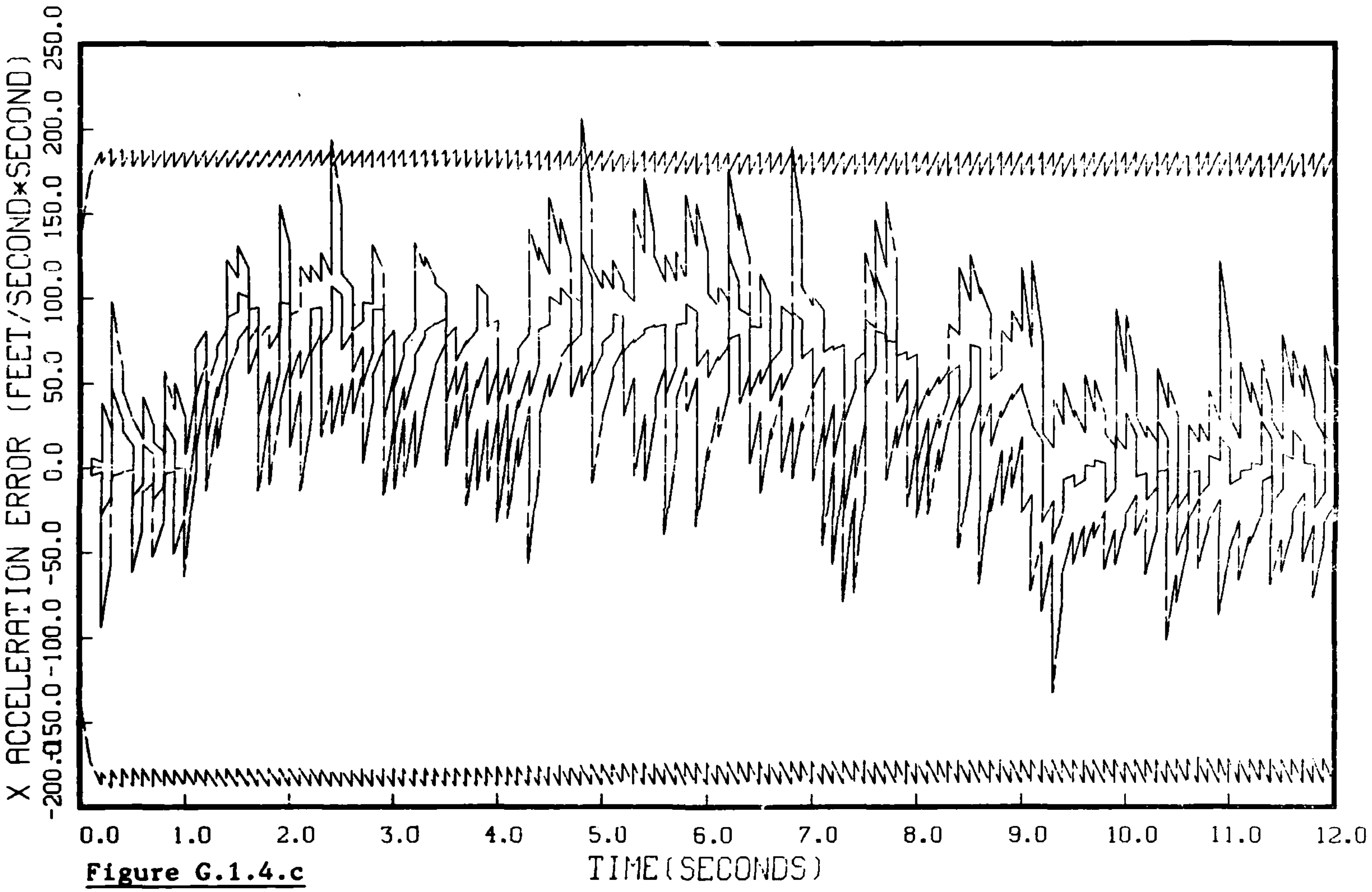


Figure G.1.4.c

STATE 3, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

4 SOFEPL DATE AND TIME - 85/10/01. 17.08.31. SOFE DATE AND TIME - 85/10/01. 16.33.53. WPAFB, DISSPLA. SOFEPL VERSION 2.3

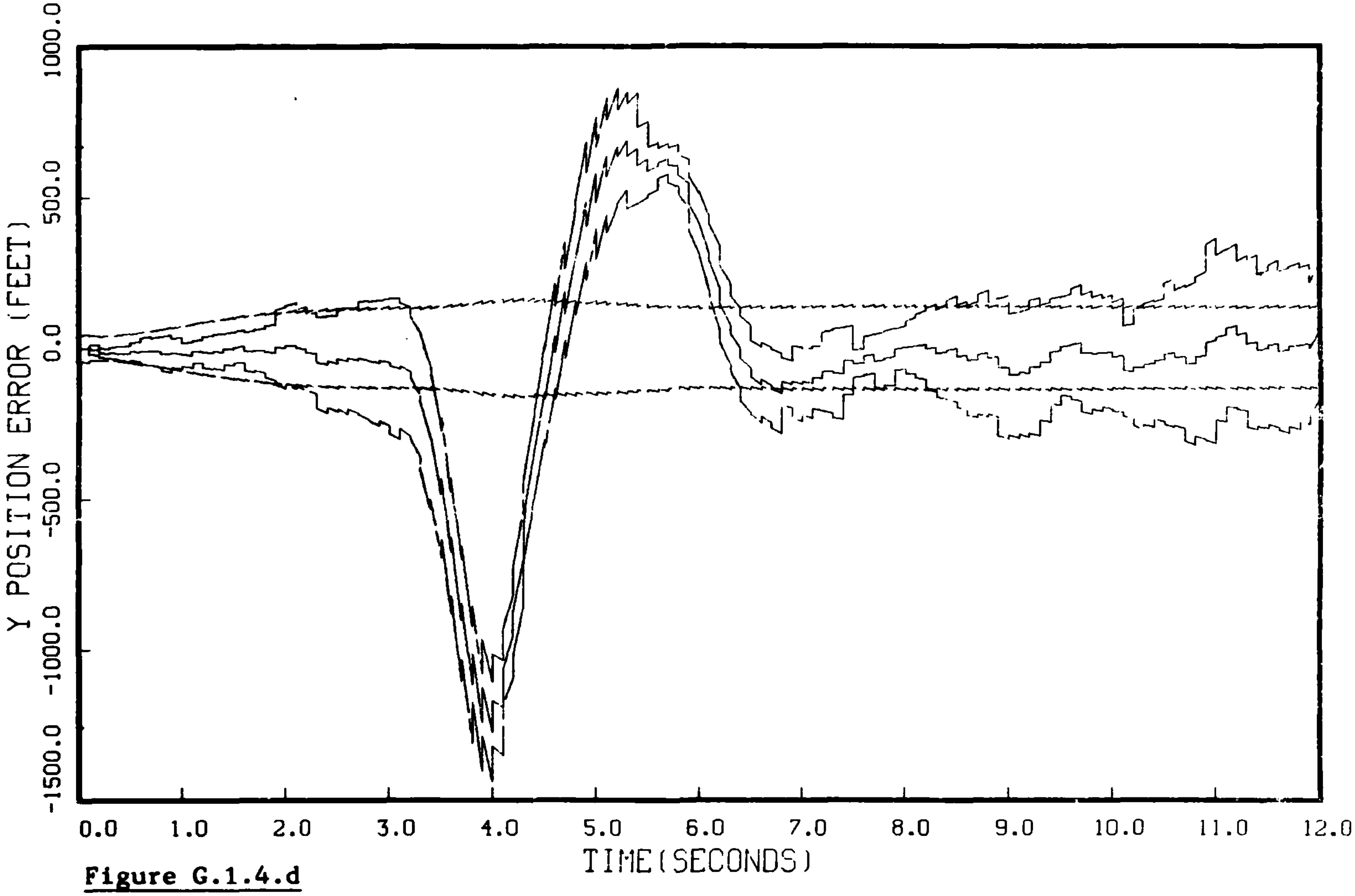


Figure G.1.4.d

STATE 4, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

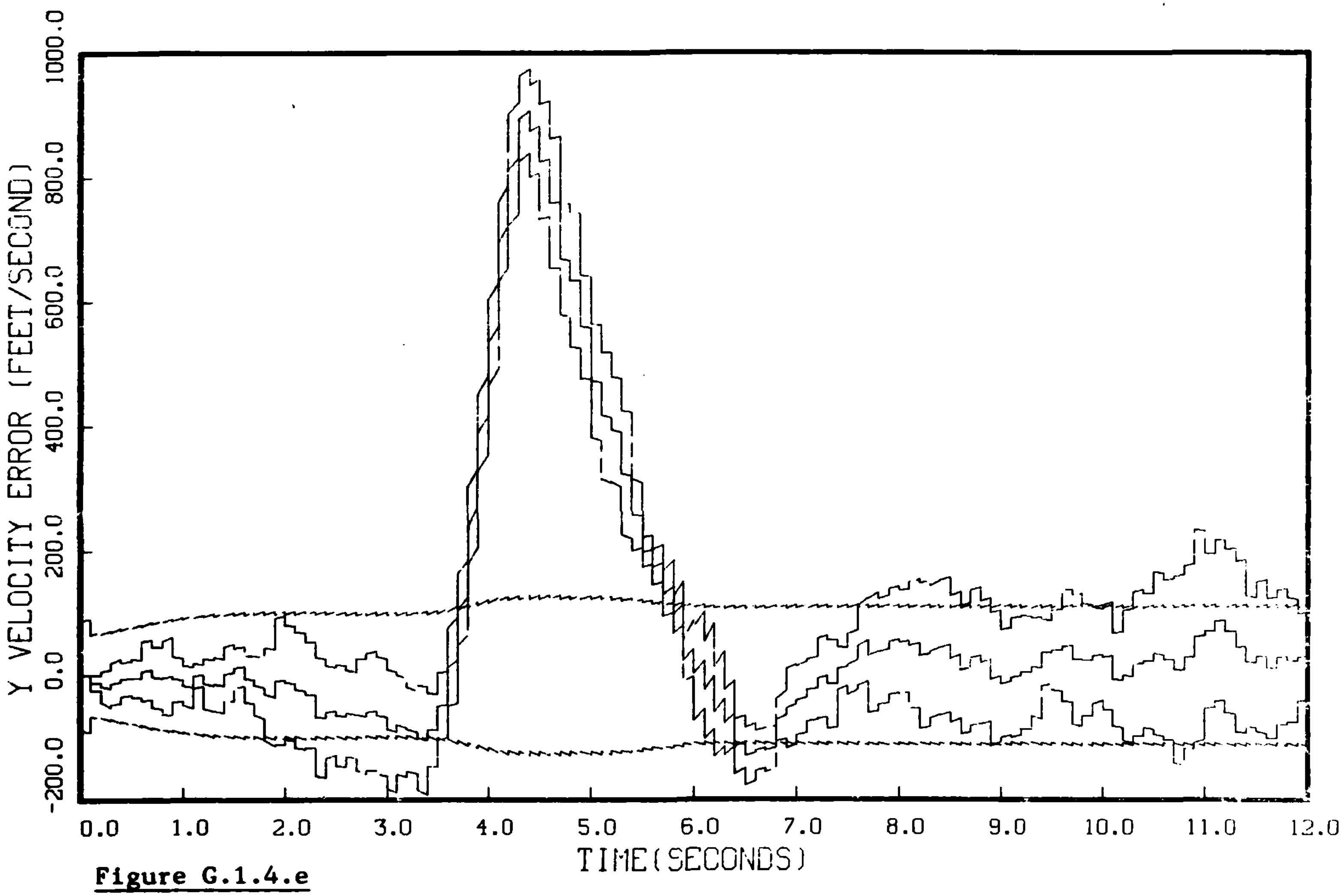


Figure G.1.4.e

STATE 5, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

6 SOFEPL DATE AND TIME - 05/10/01. 17.08.31. SOFE DATE AND TIME - 05/10/01. 16.33.53. WPAFB, DISSPLA, SOFEPL VERSION 2.2

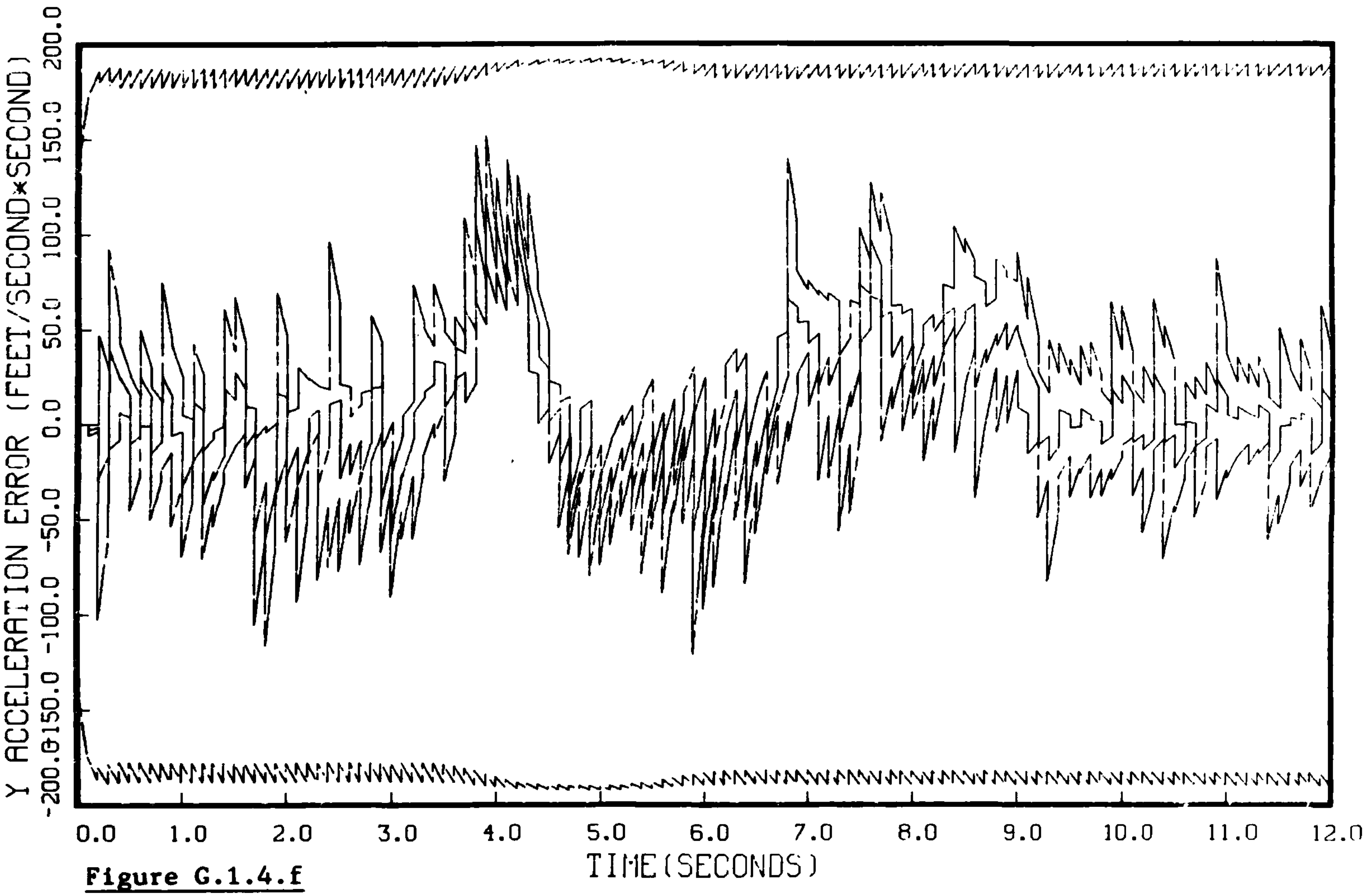


Figure G.1.4.f

STATE 6, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

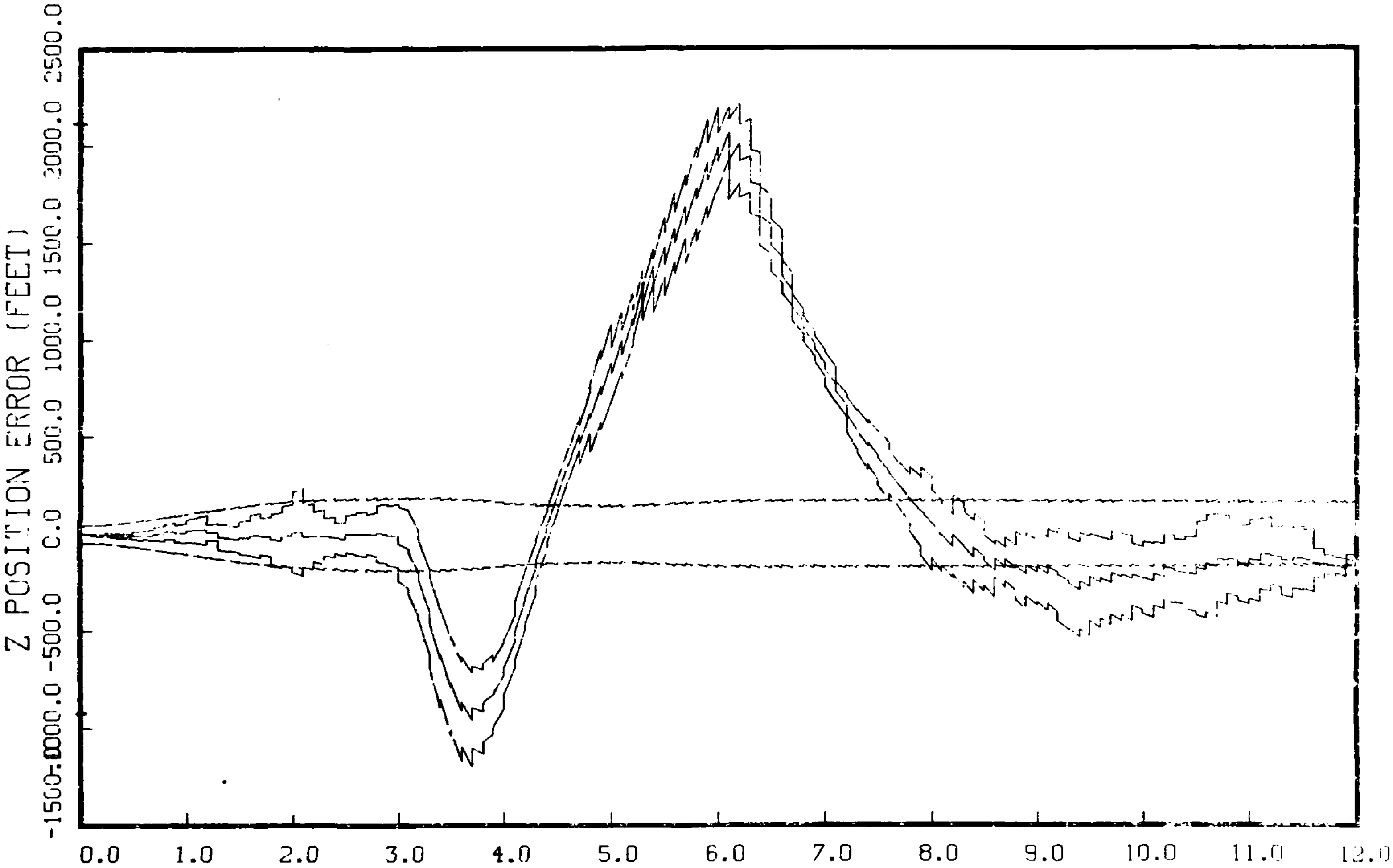


Figure G.1.4.g

STATE 7, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.'. UPDATE=.1, 5 RUNS

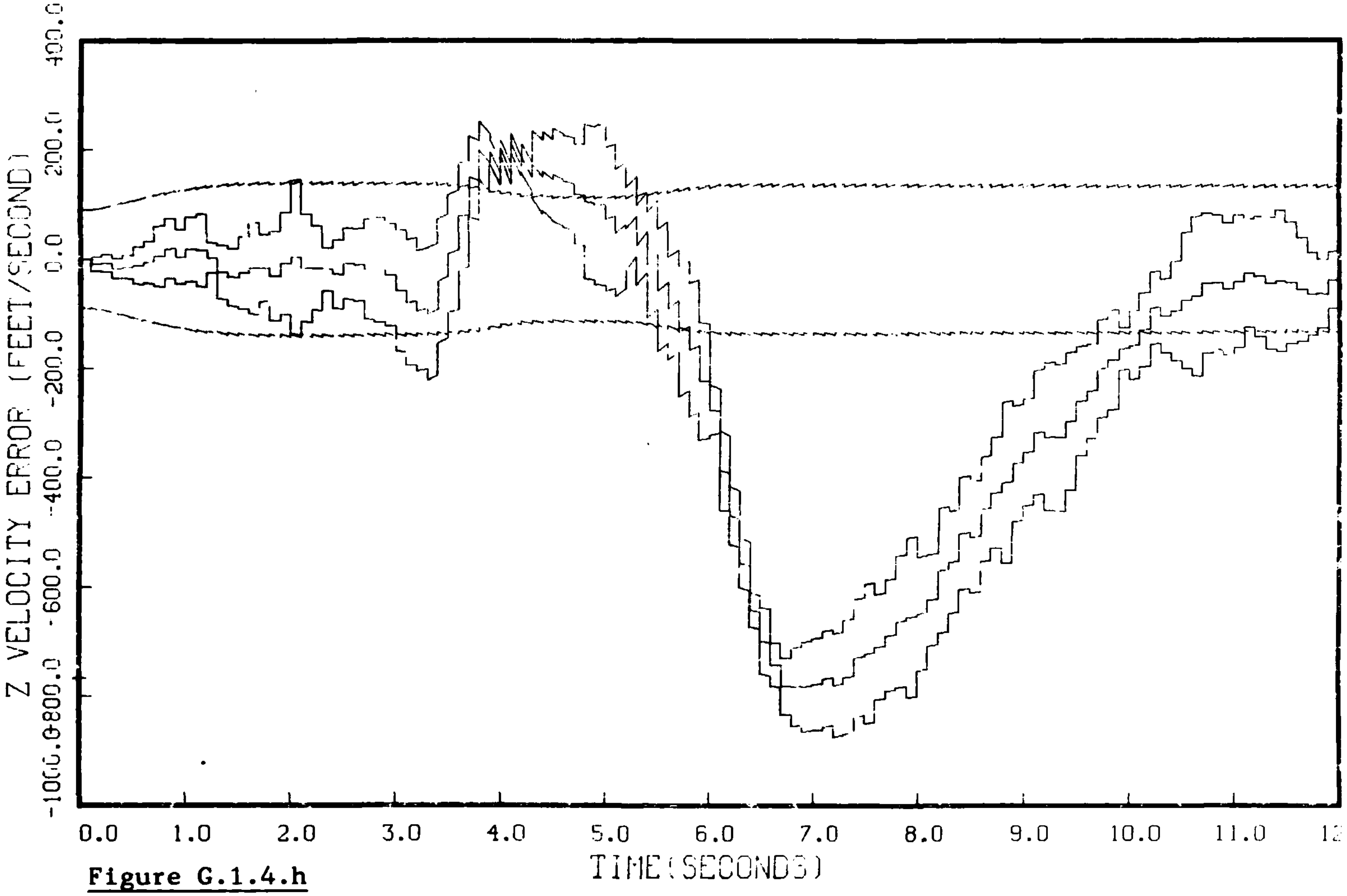


Figure G.1.4.h

STATE 8, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE=40.000.', UPDATE=.1, 5 RUNS

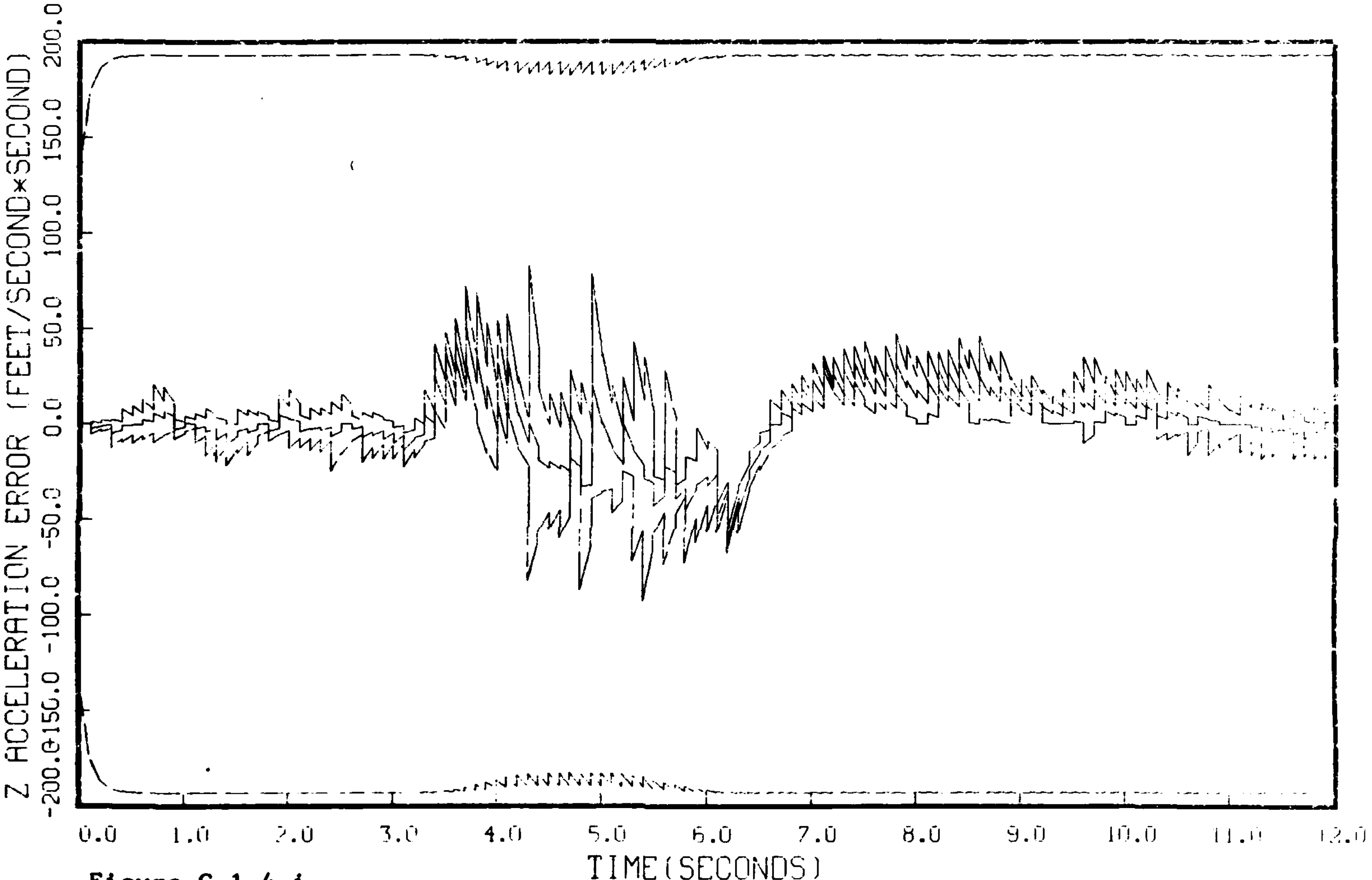


Figure G.1.4.i

STATE 9, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APG 120, BEAM ATTACK, INITIAL RANGE 10,000.', UPDATE-.1, 5 RUNS

63 PS1 5 1 SOFEPL DATE AND TIME - 85/10/01. 17.58.43. SOFE DATE AND TIME - 85/10/01. 16.41.37. WPAFB, DISSPLA, SOFEPL VERSION 2.2

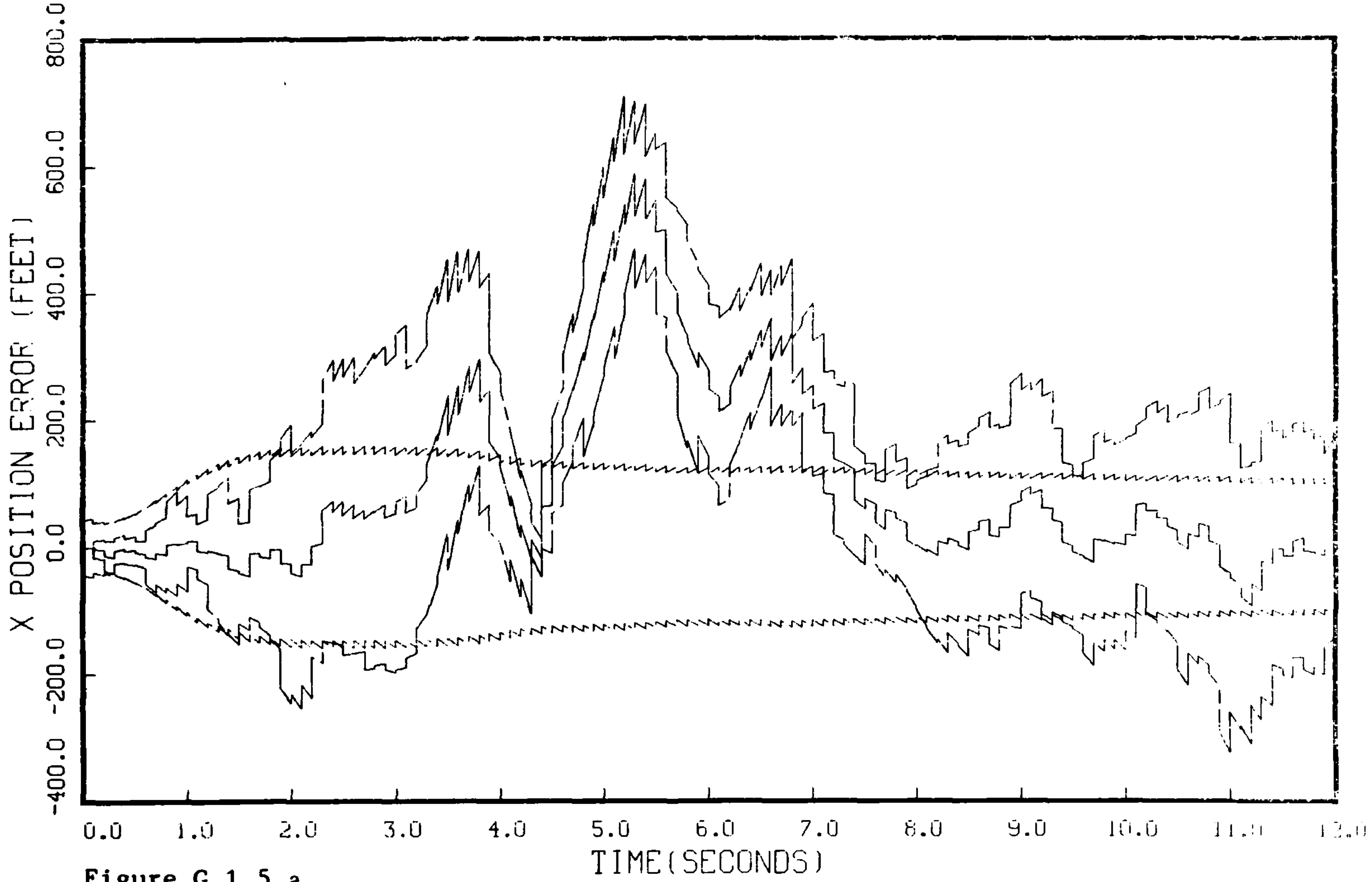


Figure G.1.5.a

STATE 1, Q(1)-Q(2)-Q(3)-3732500., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
AFG 120, BEAM ATTACK, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

2 SOFEPL DATE AND TIME - 85/10/01. 17.58.43. SOFE DATE AND TIME - 85/10/01. 16.41.37. WPAFB, DISSPLA, SOFEPL VERSION

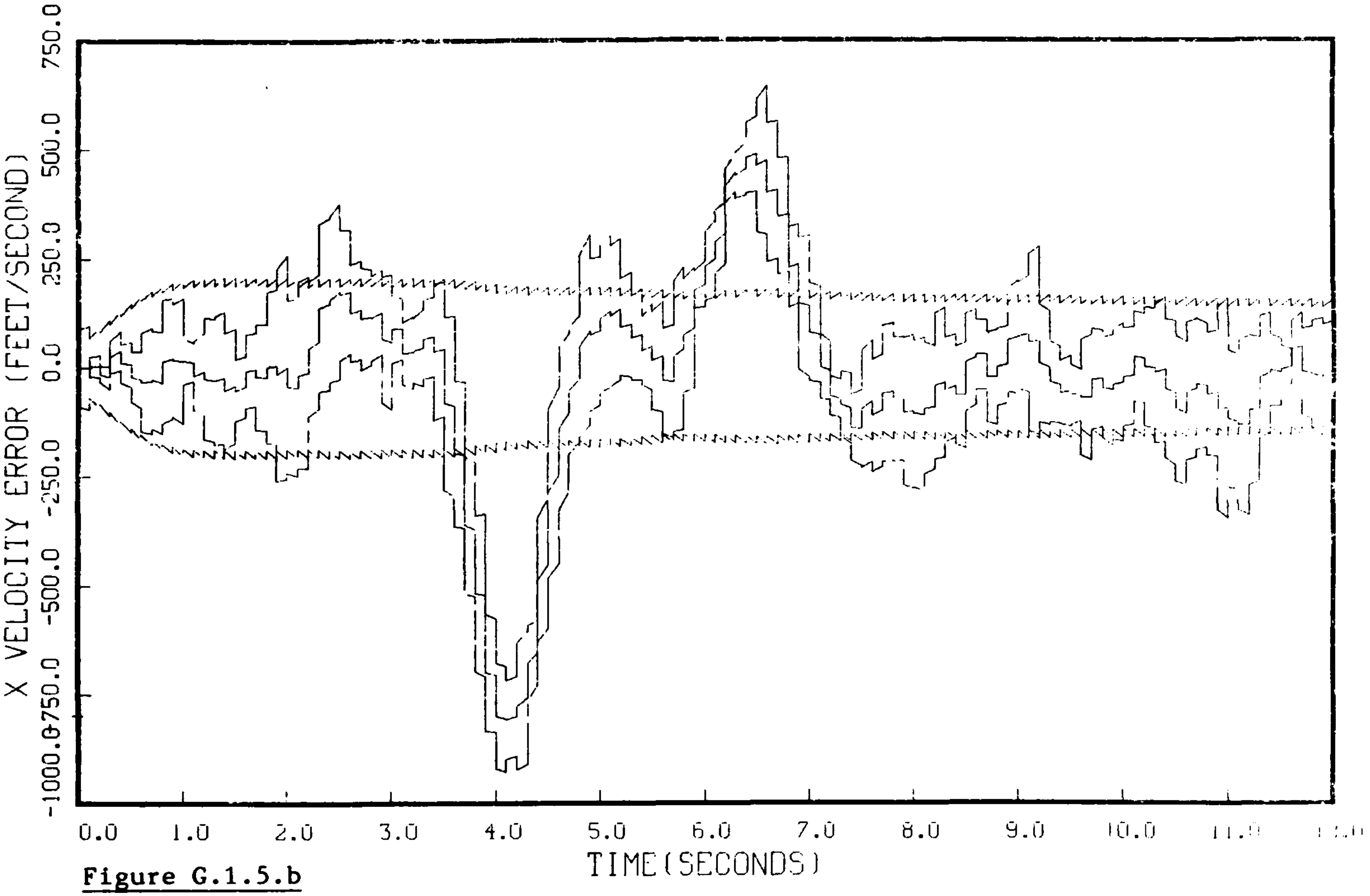


Figure G.1.5.b

STATE 2, Q(1)-Q(2)-Q(3)-3732500., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
APC 125, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

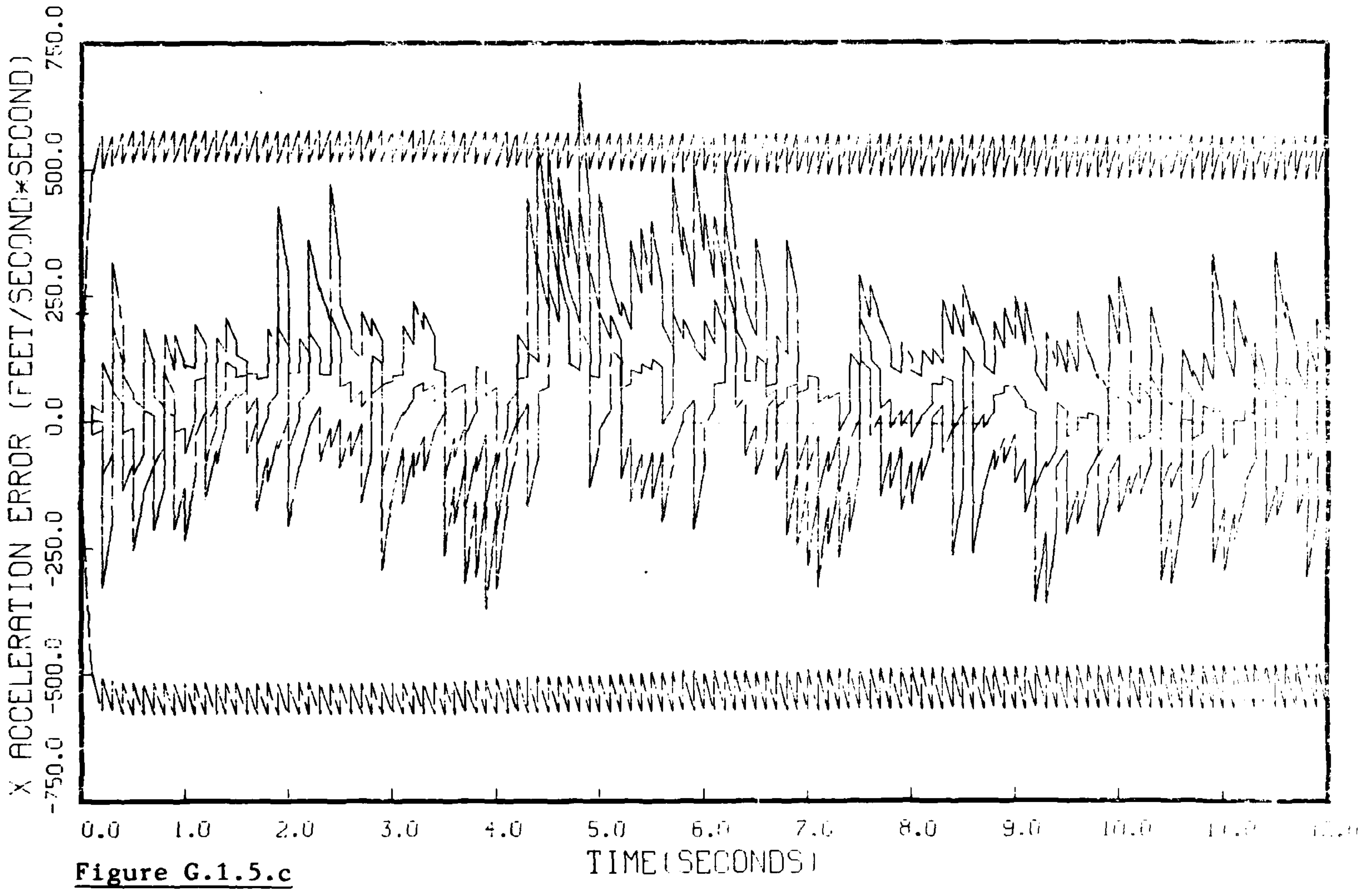


Figure G.1.5.c

STATE 3, Q(1)=Q(2)=Q(3)=3732500., TAU(1)=.2,TAU(2-3)=.2, ALL MEHS
APG 120, BEAM ATTACK, INITIAL RANGE 10,000. , UPDATE .1, 5 RUNS

4 SOFEPL DATE AND TIME - 85/10/01. 17.58.43. SOFE DATE AND TIME - 85/10/01. 16.41.37. WPAFB, DISSPLA, SOFEPL VERSION 2.2

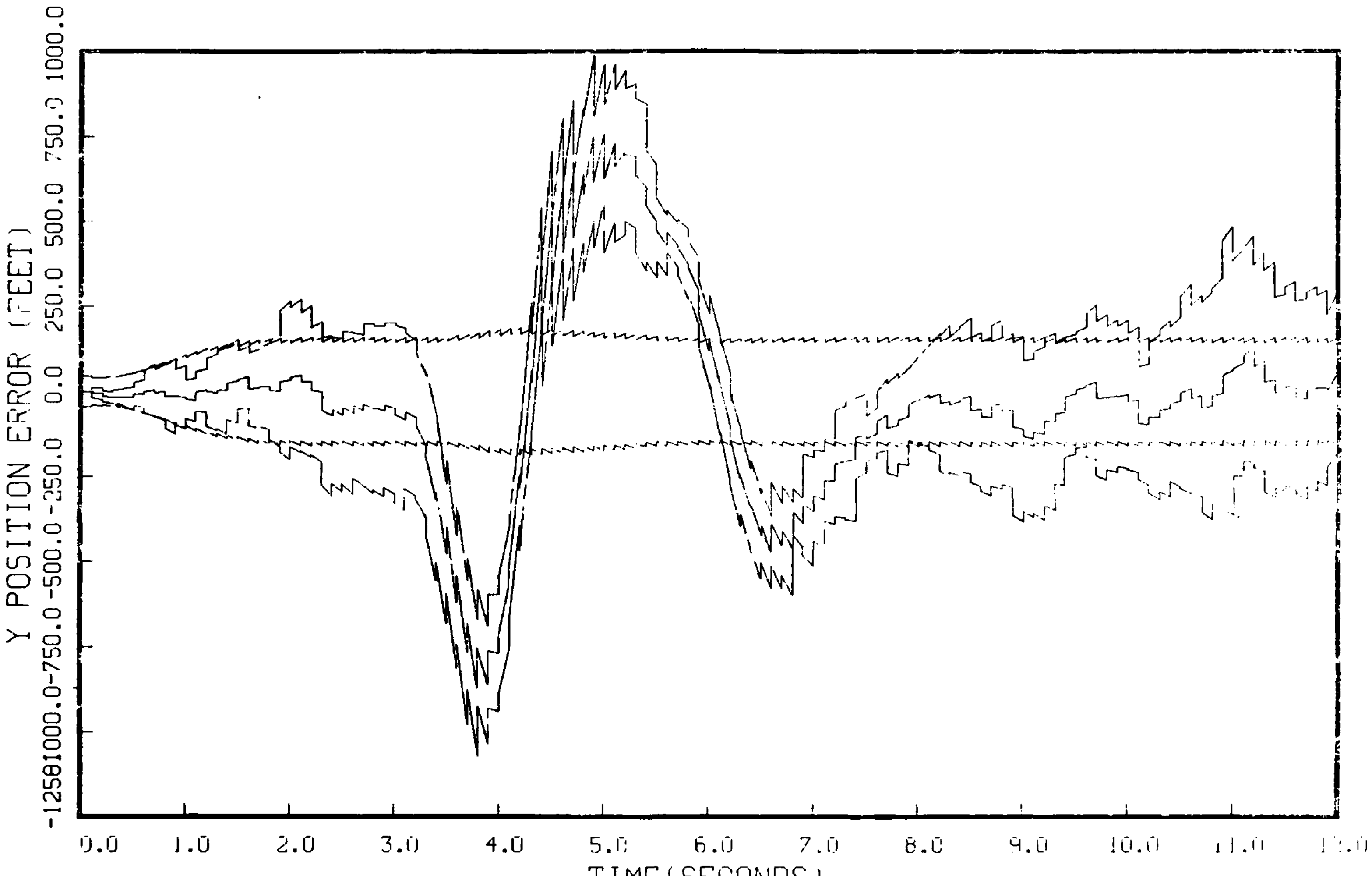


Figure G.1.5.d

STATE 4, Q(1)-Q(2)-Q(3)-3732500.,' TAU(1)-.2,TAU(2-3)=.2, ALL MEAS
APC 120, BEAM ATTACK, INITIAL RANGE 40,000.', UPDATE-.1, 5 RUNS

5 SUPEPL DATE AND TIME - 85/10/01. 17.56.43. SUPL DATE AND TIME - 85/10/01. 16.41.37. WPAFB, DISSPLA. SUPEPL VERSION 2.4

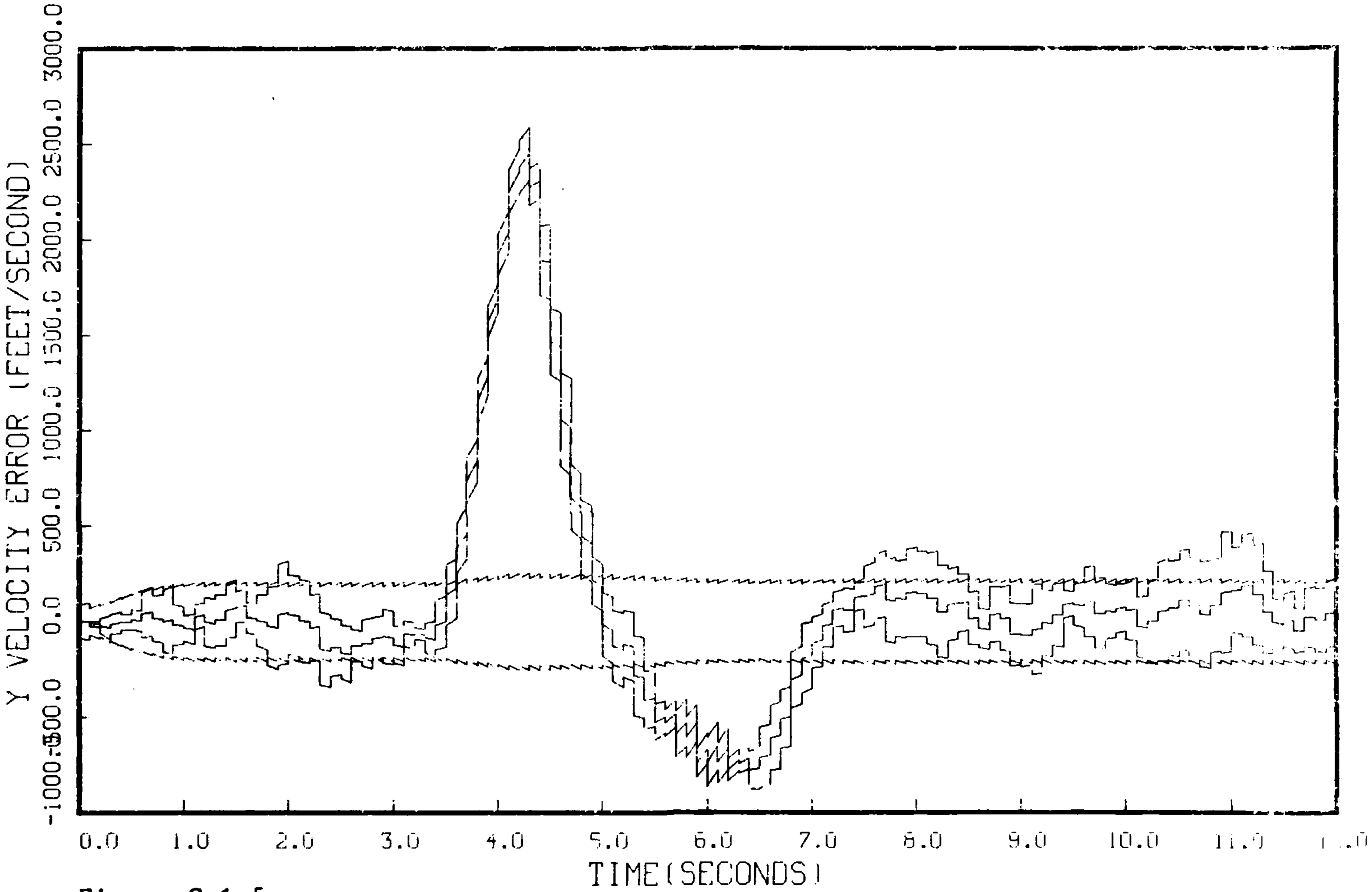


Figure G.1.5.e
STATE 5, Q(1)=Q(2)=Q(3)=3732500., TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
ARO 120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

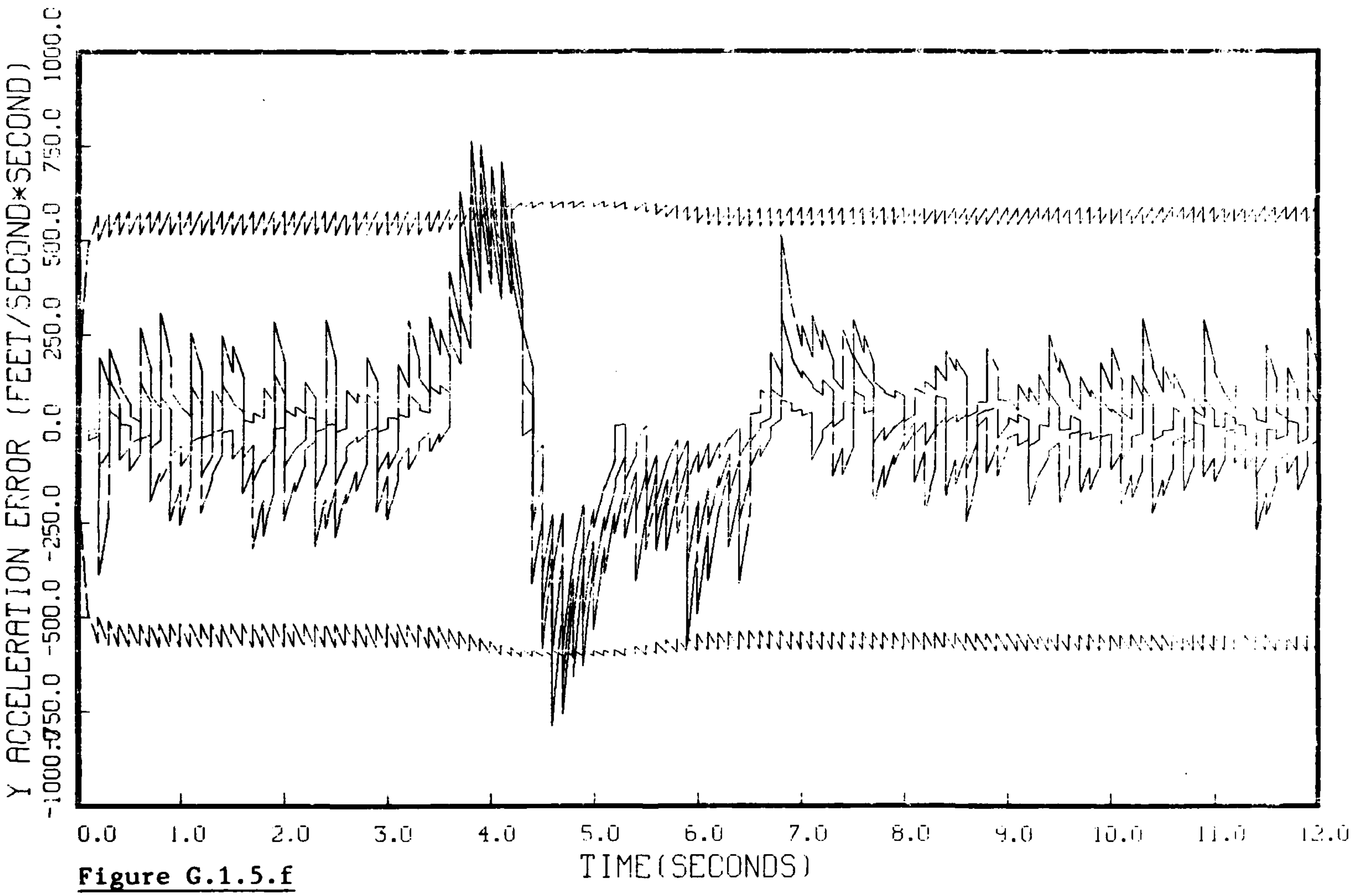


Figure G.1.5.f

STATE 6, Q(1)=Q(2)=Q(3)=3732500., TAU(1)=.2,TAU(2-3)=.2, ALL MEAS
APG 120, BEAM ATTACK, INITIAL RANGE 10,000.', UPDATE=.1, 5 RUNS

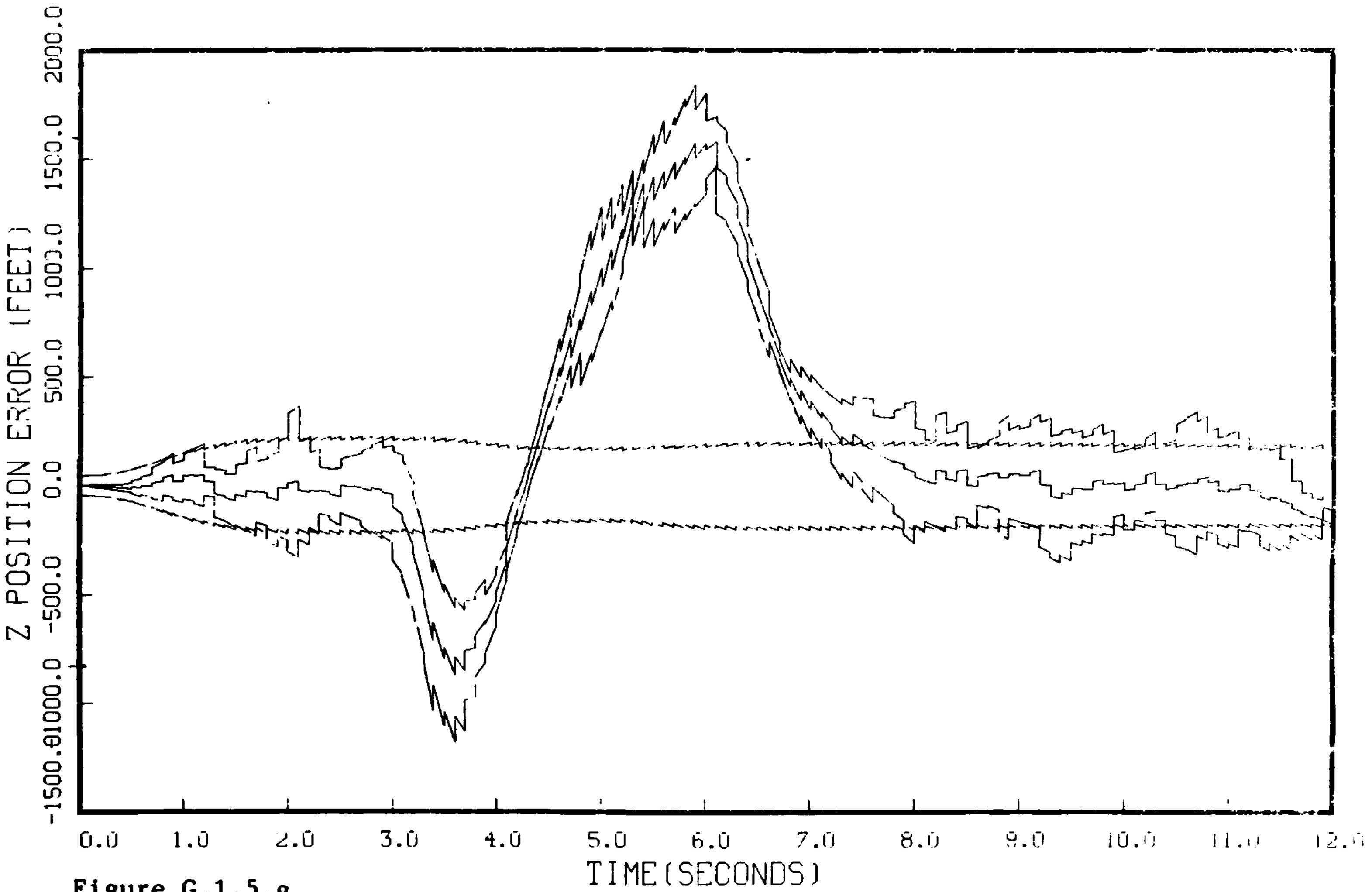


Figure G.1.5.g

STATE 7, Q(1)-Q(2)-Q(3)-3732500., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
AFC 120, BEAM ATTACK, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

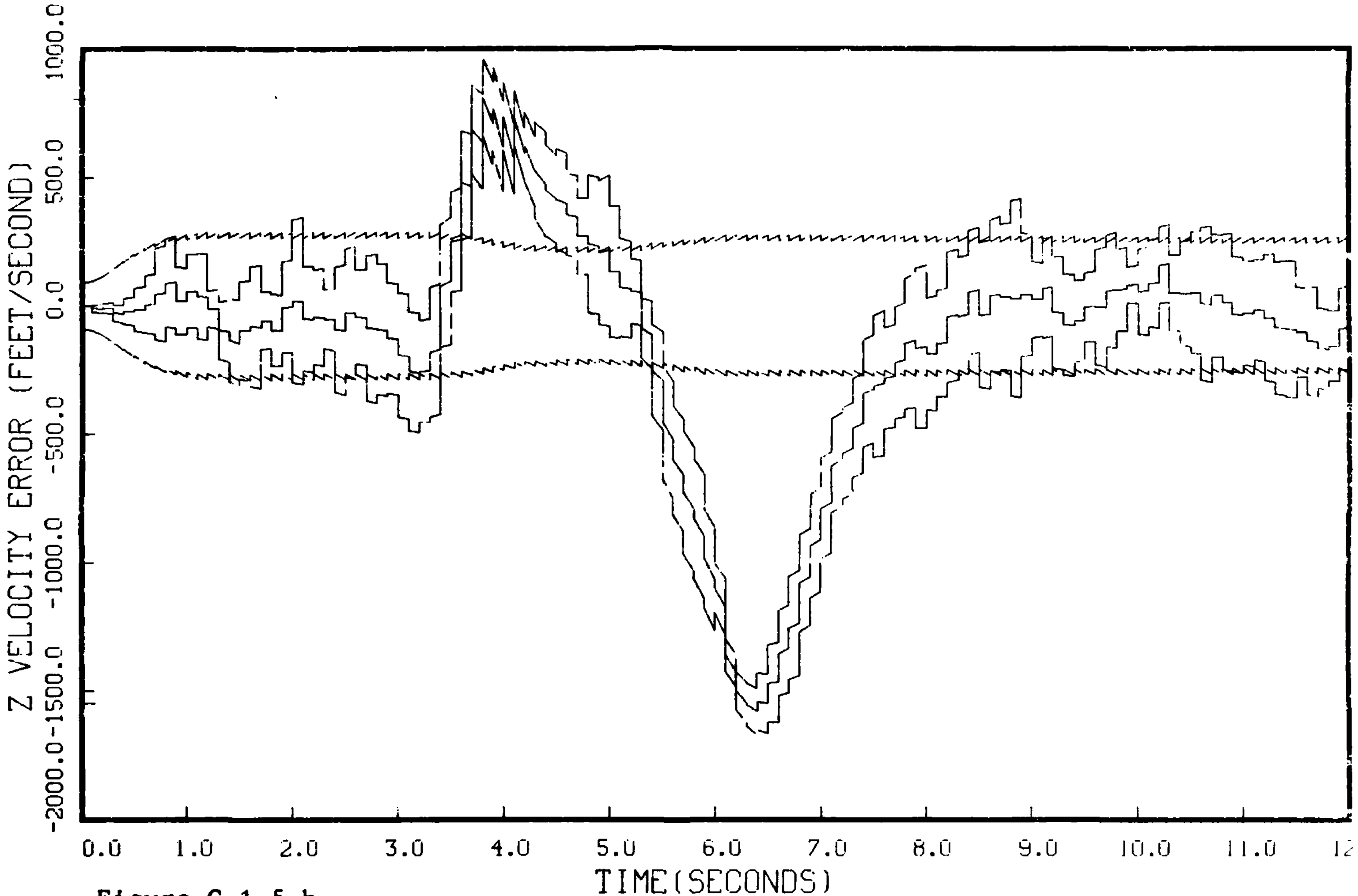


Figure G.1.5.h

STATE 8, Q(1)-Q(2)-Q(3)-3732500., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
RFG 120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

9 SUFEPL DATE AND TIME - 85/10/01. 17.58.4?. SOFE DATE AND TIME - 85/10/01. 16.41.37. WPAFB, DISSPLA, SOFEPL VERSION 2.2

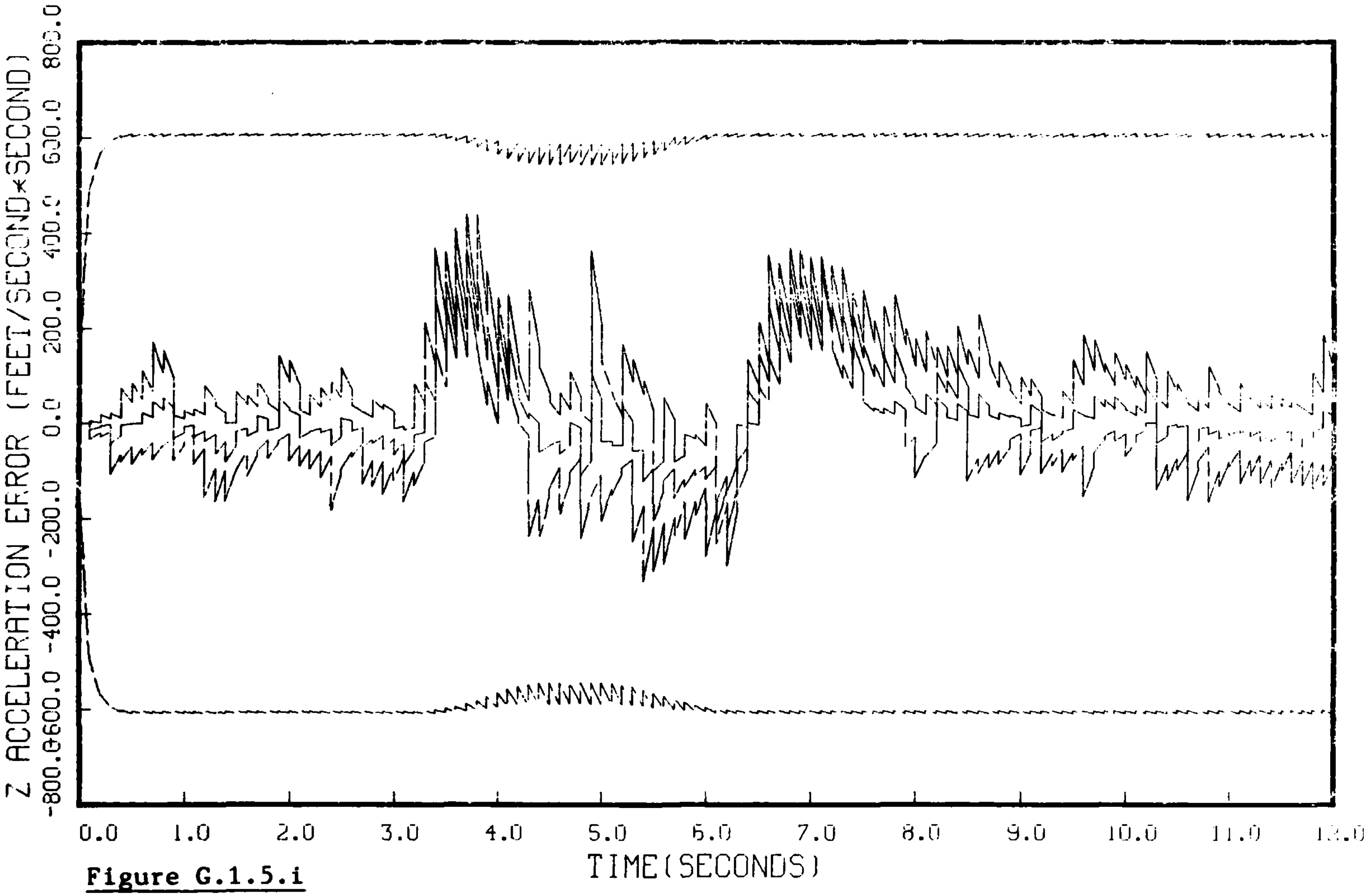


Figure G.1.5.i

STATE 9, Q(1)-Q(2)-Q(3)-3732500., TAU(1)-.2,TAU(2-3)-.2, ALL MEAS
AFO 120, BEAM ATTACK, INITIAL RANGE 40,000.', UPDATE-.1, 5 RUNS

66 PS2 1 1 SOFEPL DATE AND TIME - 85/10/02. 16.28.01. SOFE DATE AND TIME - 85/10/02. 16.03.02. WPAFB, DISSPLA, SOFEPL VERSION 2.2

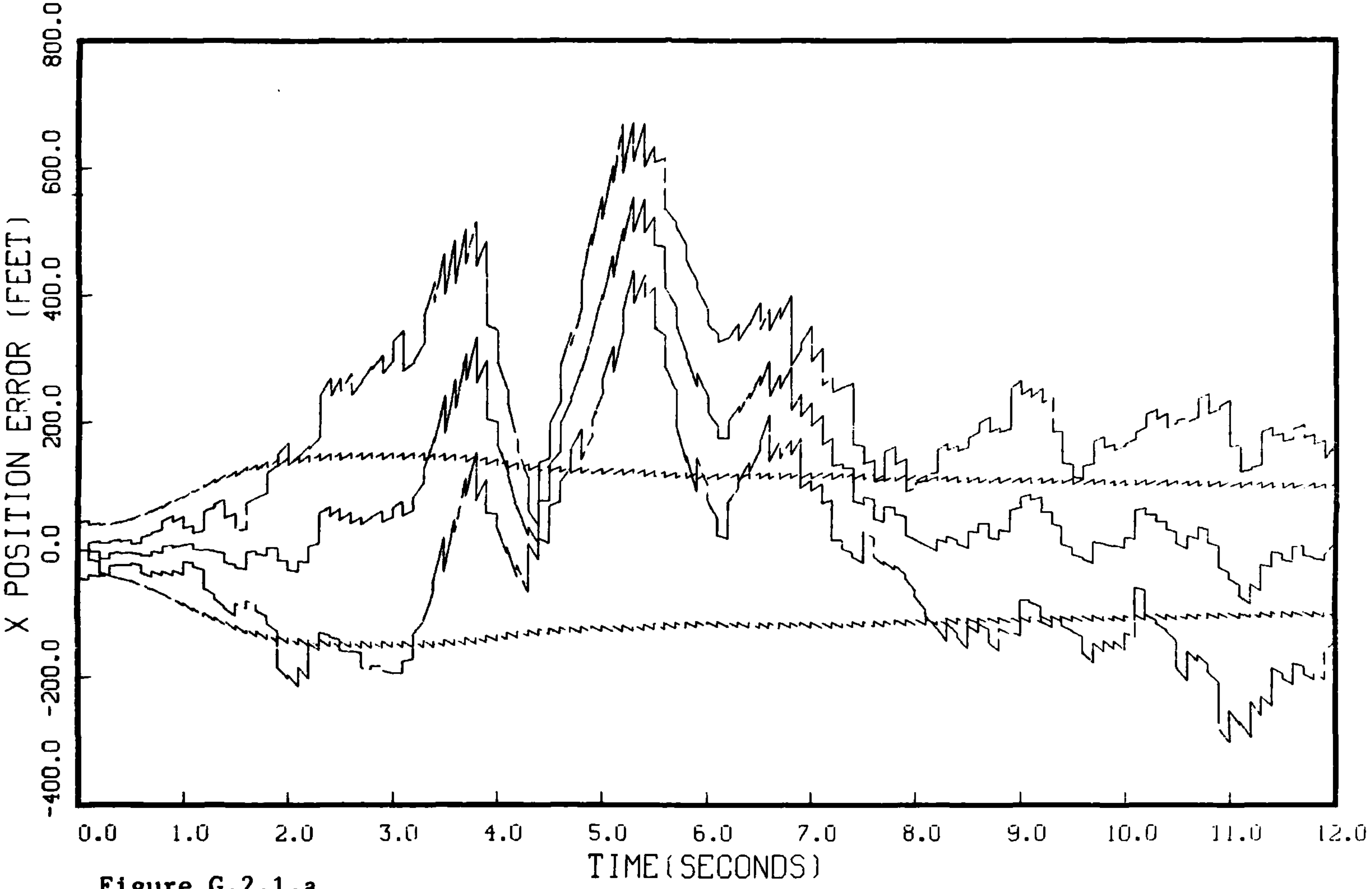


Figure G.2.1.a
STATE 1, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

2 SOFEPL DATE AND TIME - 85/10/02. 16.28.01. SOFE DATE AND TIME - 85/10/02. 16.03.02. WPAFB, DISSPLA, SOFEPL VERSION 2.2

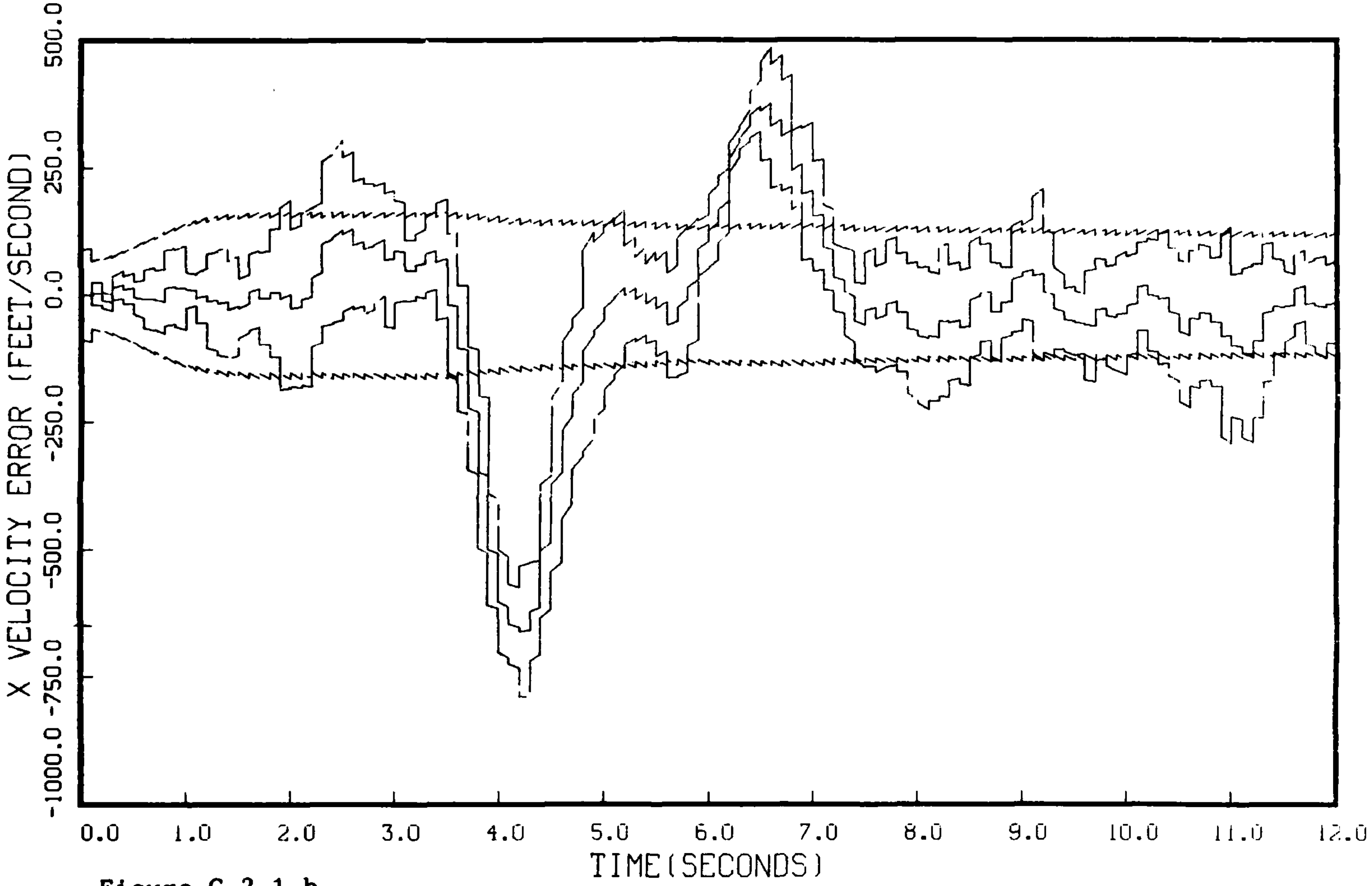


Figure G.2.1.b
STATE 2, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.5,TAU(2-3)=.5, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

3 SOFEPL DATE AND TIME - 85/10/02. 16.28.01. SOFE DATE AND TIME - 85/10/02. 16.03.02. WPAFB, DISSPLA, SOFEPL VERSION 2.2

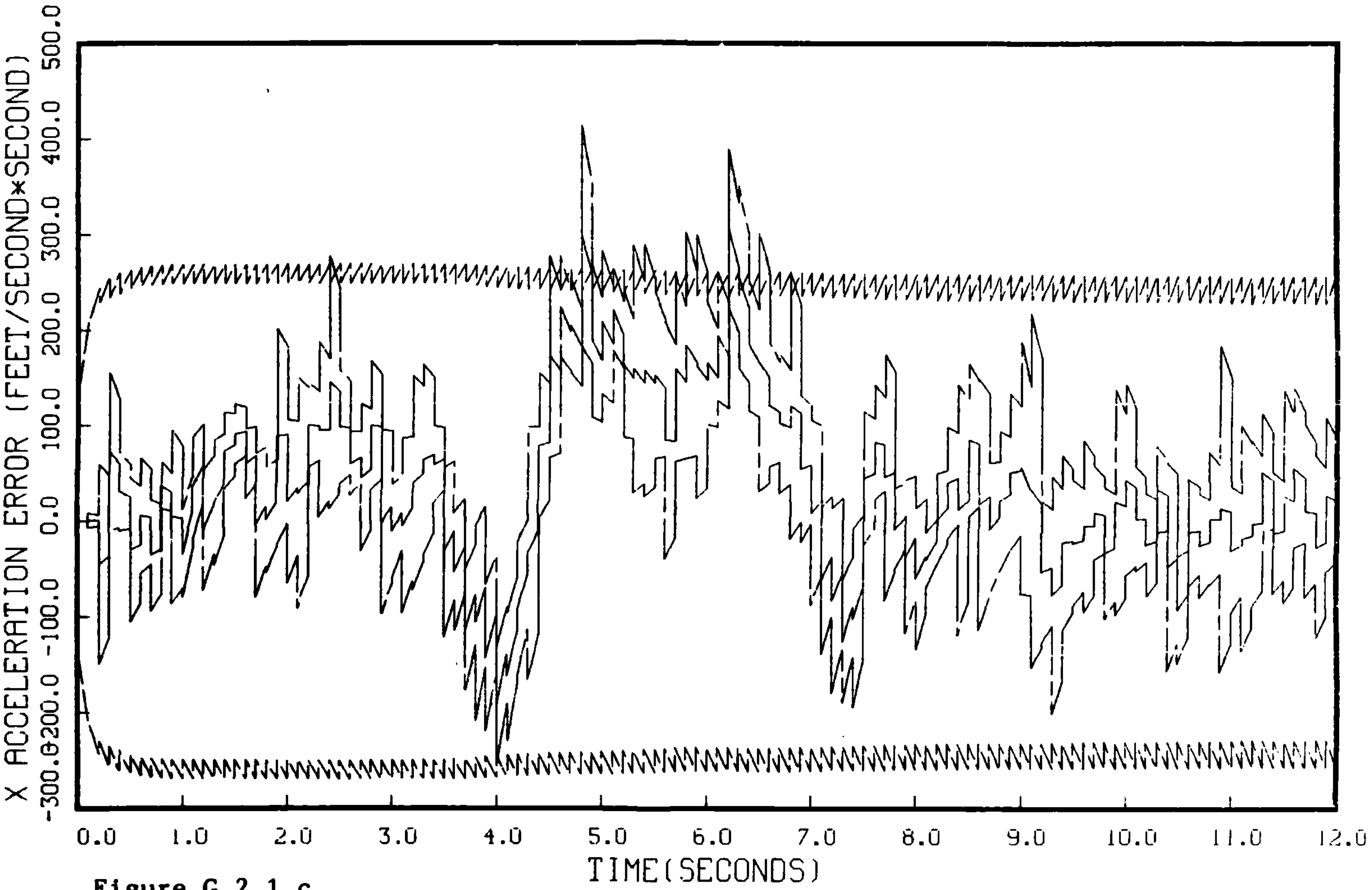


Figure G.2.1.c

STATE 3, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE-.1, 5 RUNS

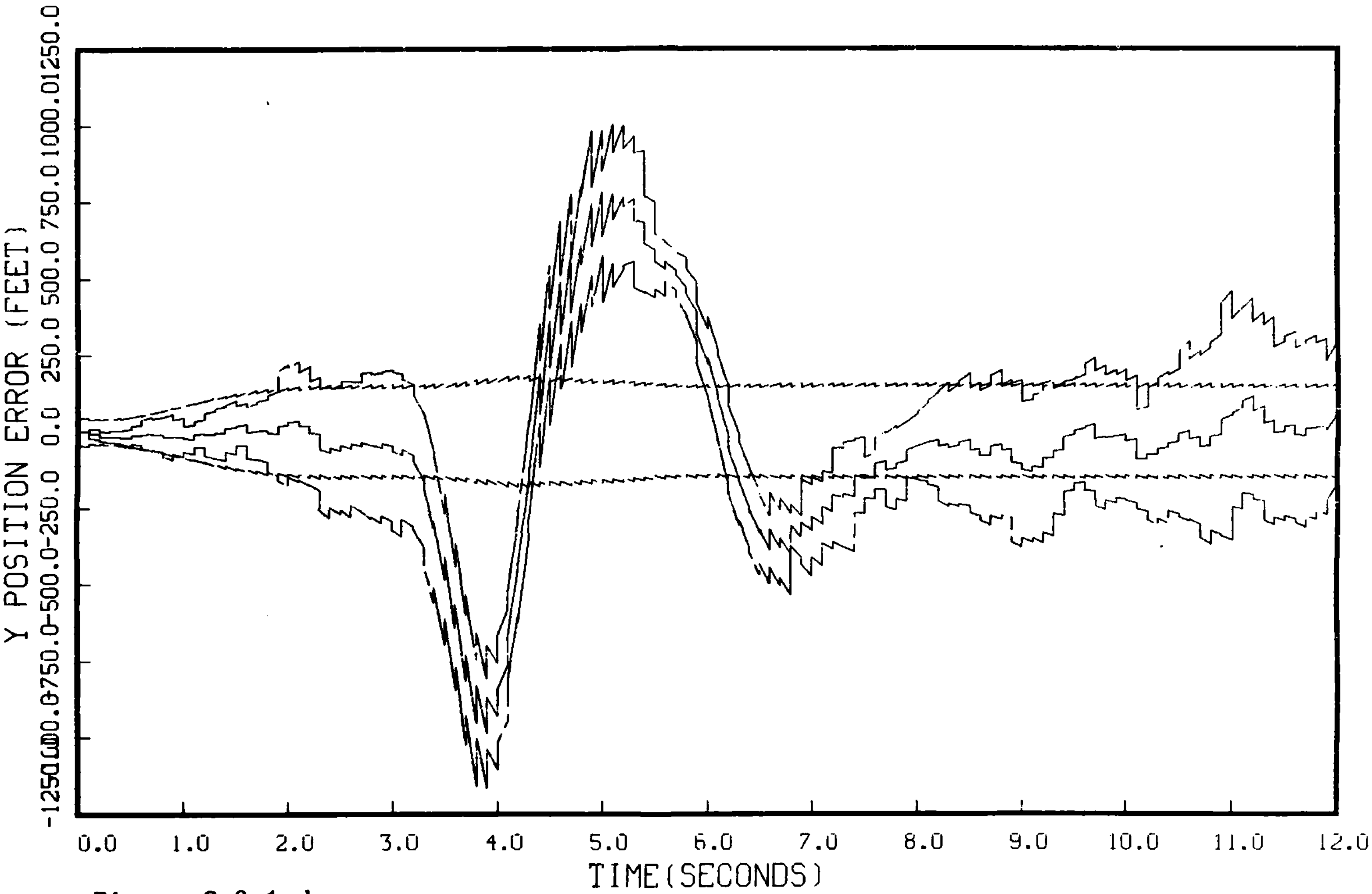


Figure G.2.1.d

STATE 4, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

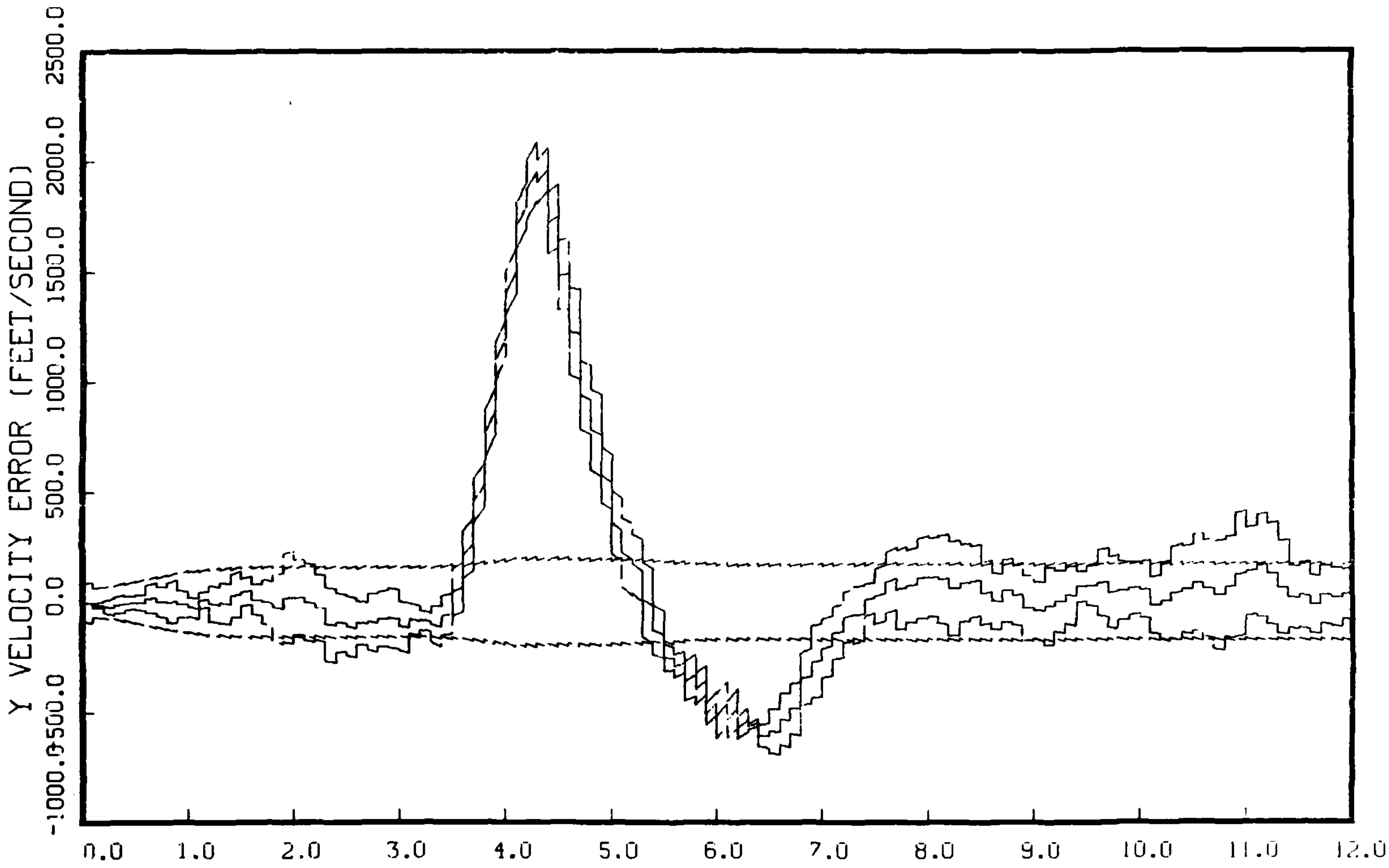


Figure G.2.1.e

STATE 5, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

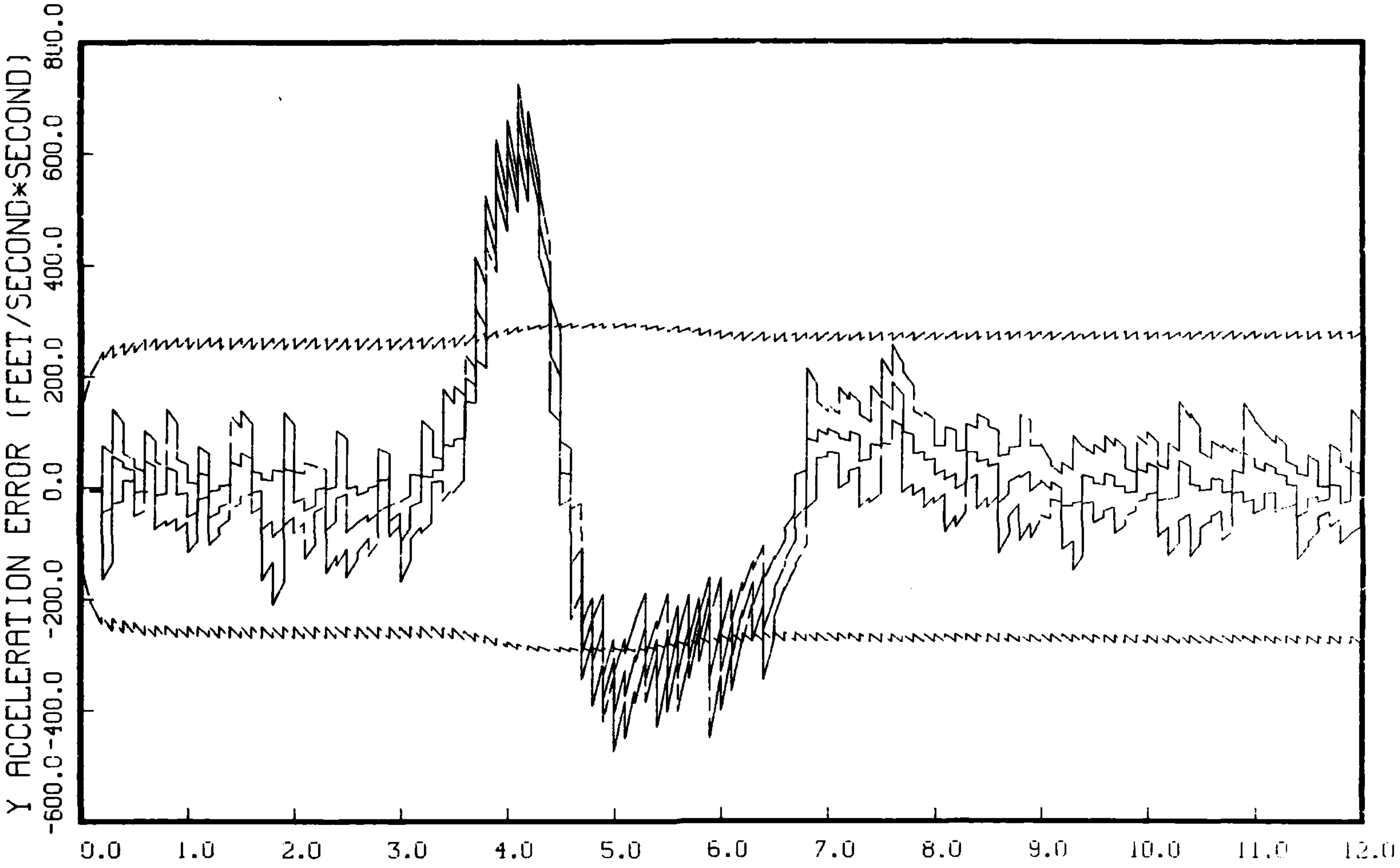


Figure G.2.1.f

STATE 6, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

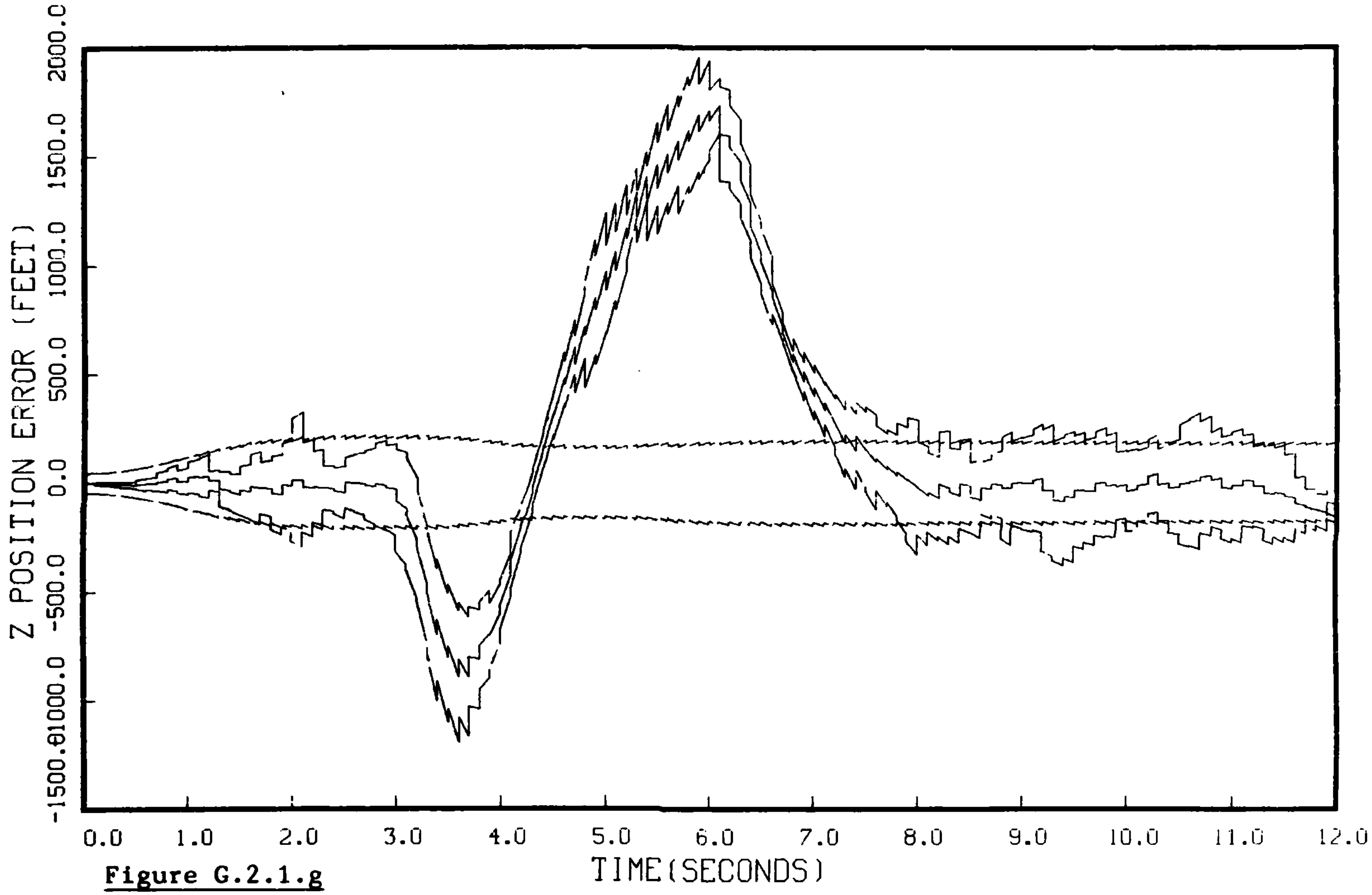


Figure G.2.1.g

STATE 7, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

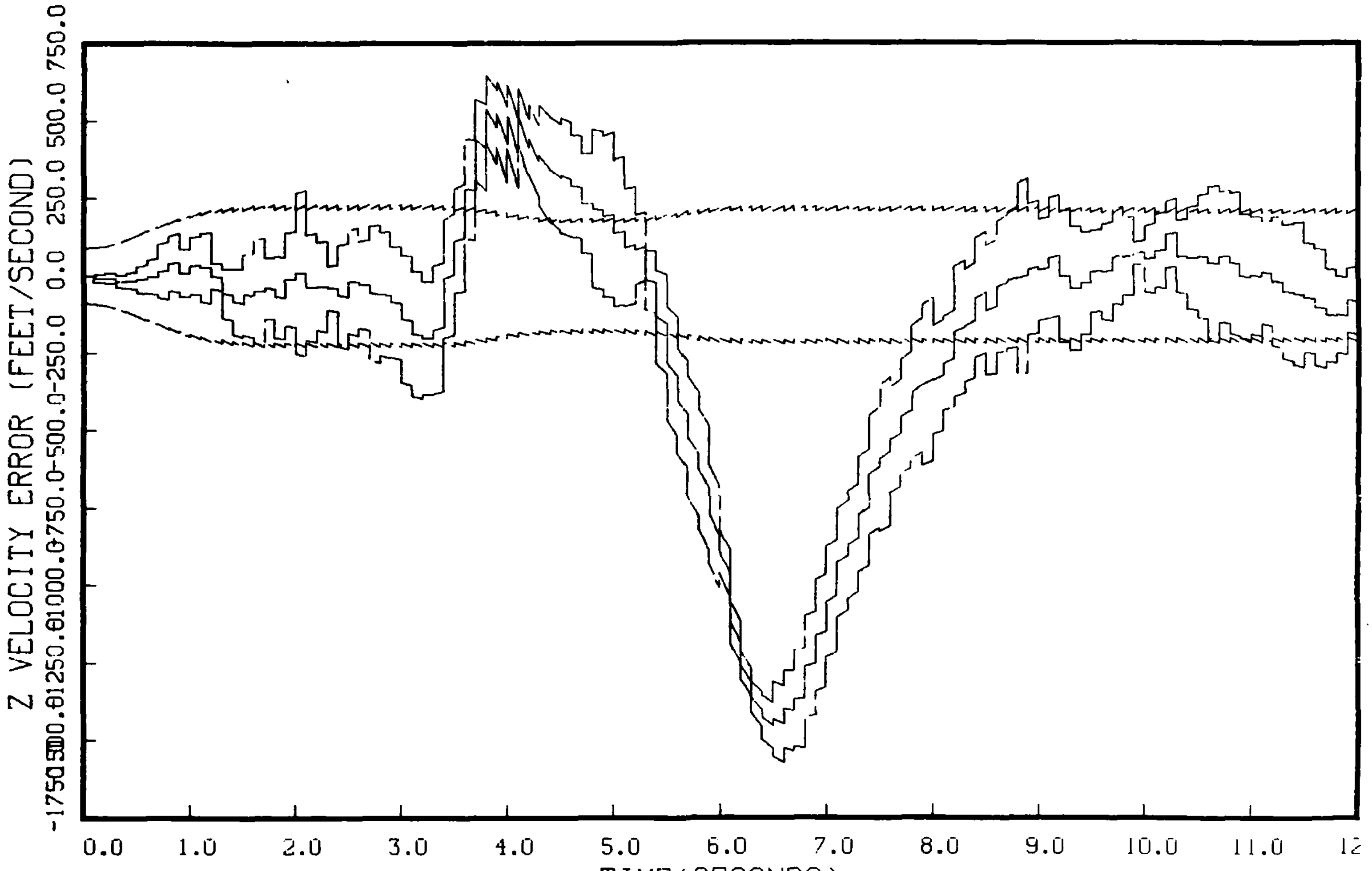


Figure G.2.1.h

STATE 8, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

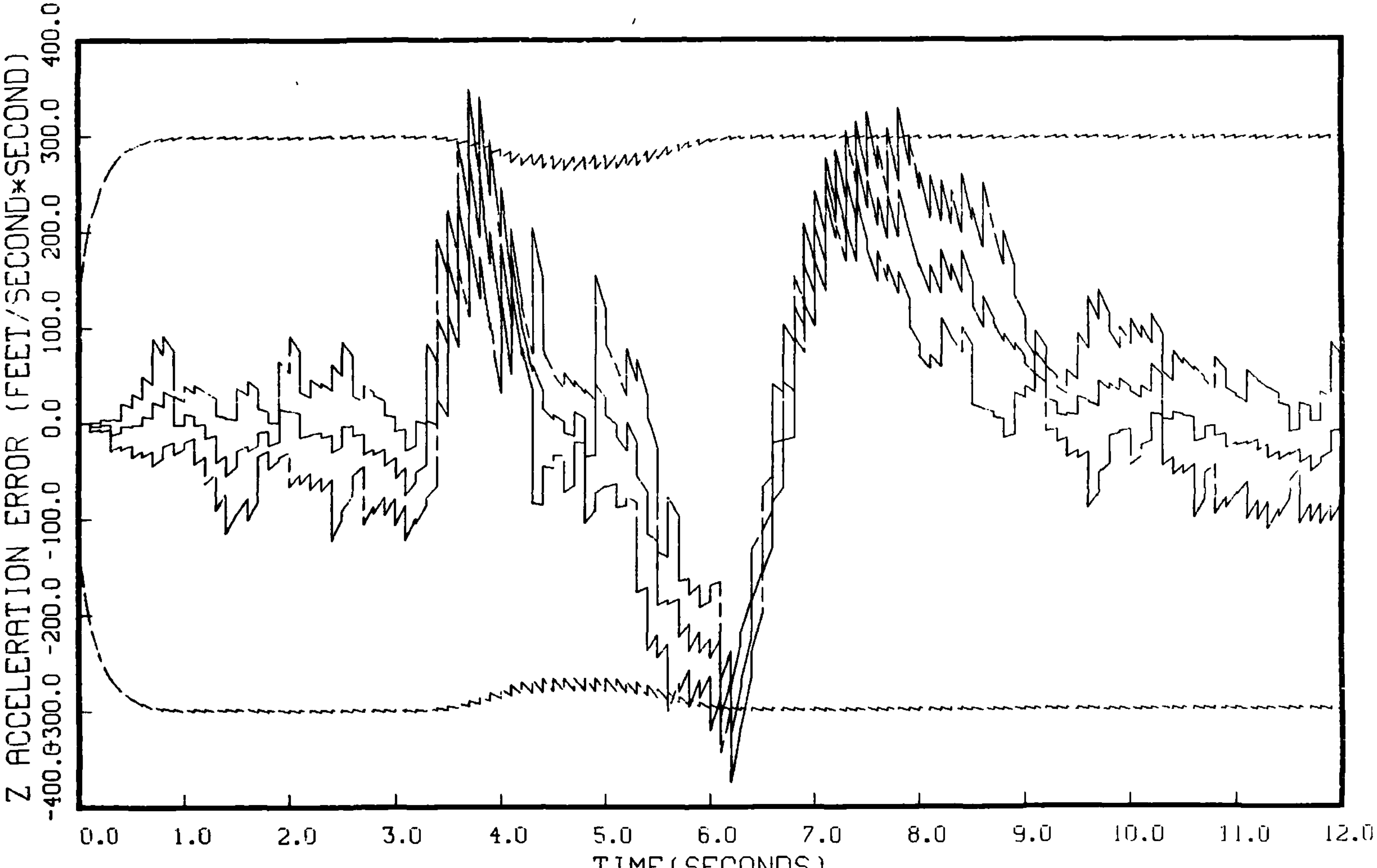


Figure G.2.1.i

STATE 9, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.5,TAU(2-3)-.5, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

Figure Set G.2.2 is the same as Figure Set G.1.5

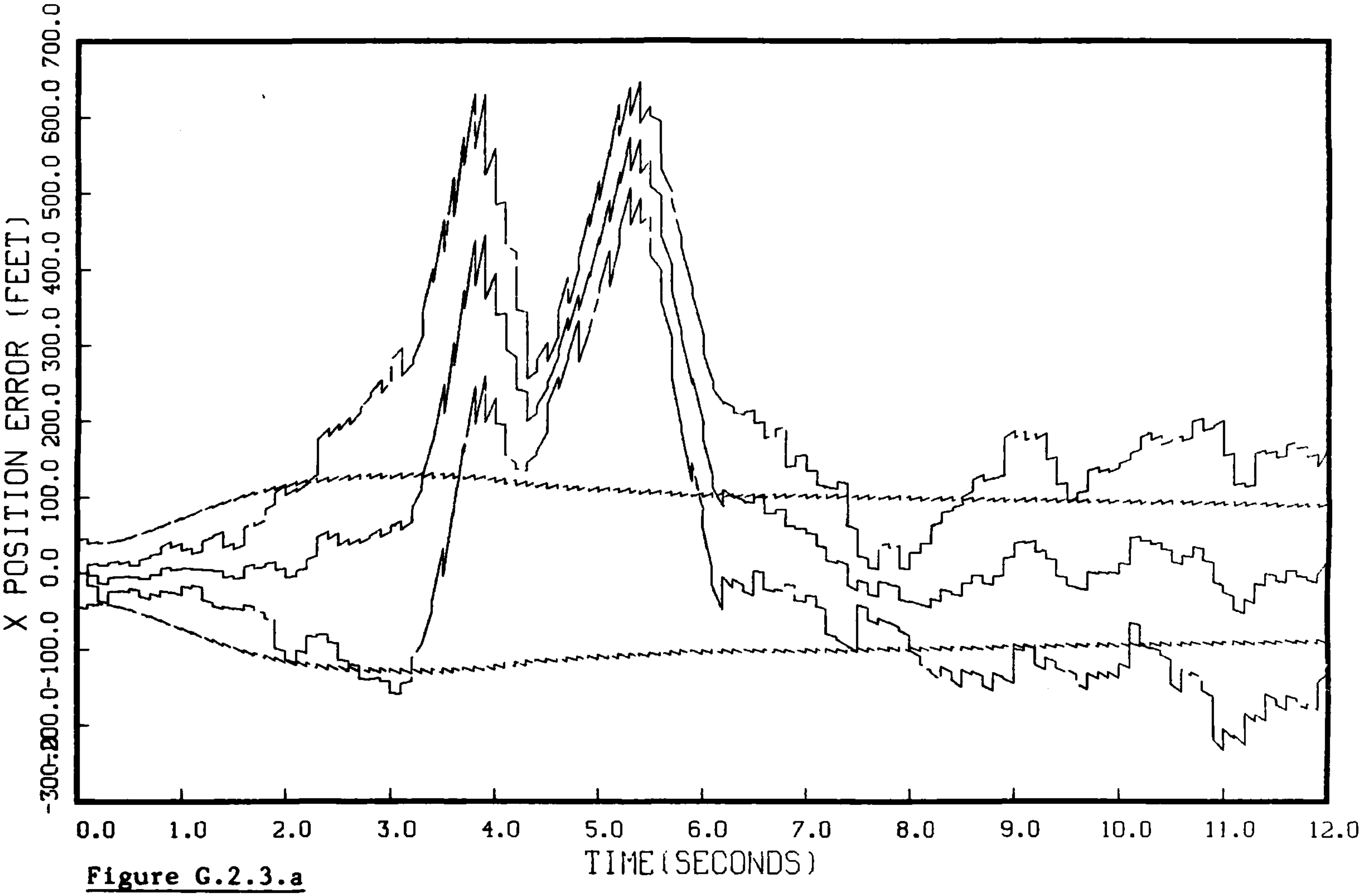


Figure G.2.3.a

STATE 1, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.167,TAU(2-3)-.167, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

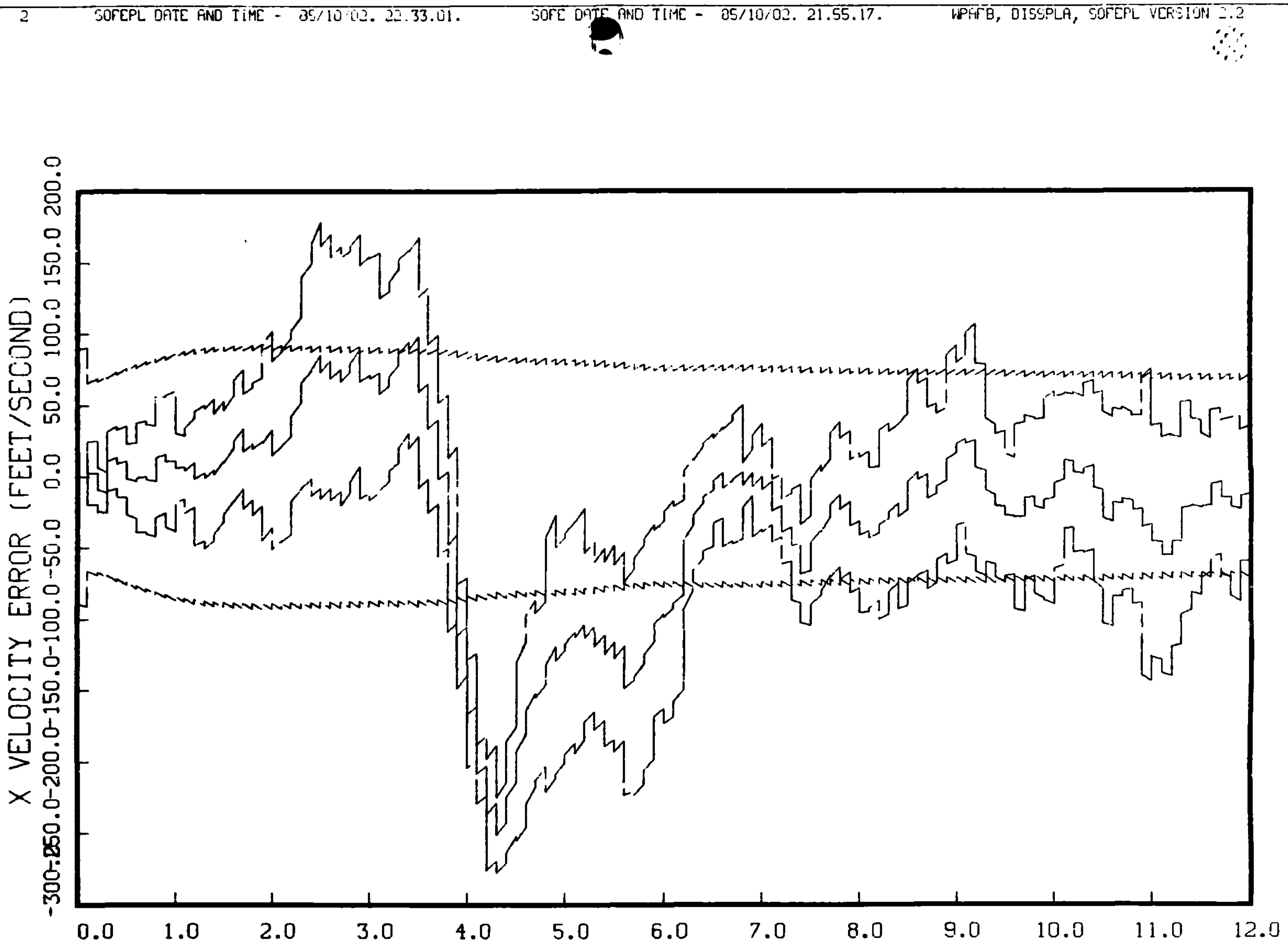


Figure G.2.3.b

STATE 2, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.167,TAU(2-3)-.167, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

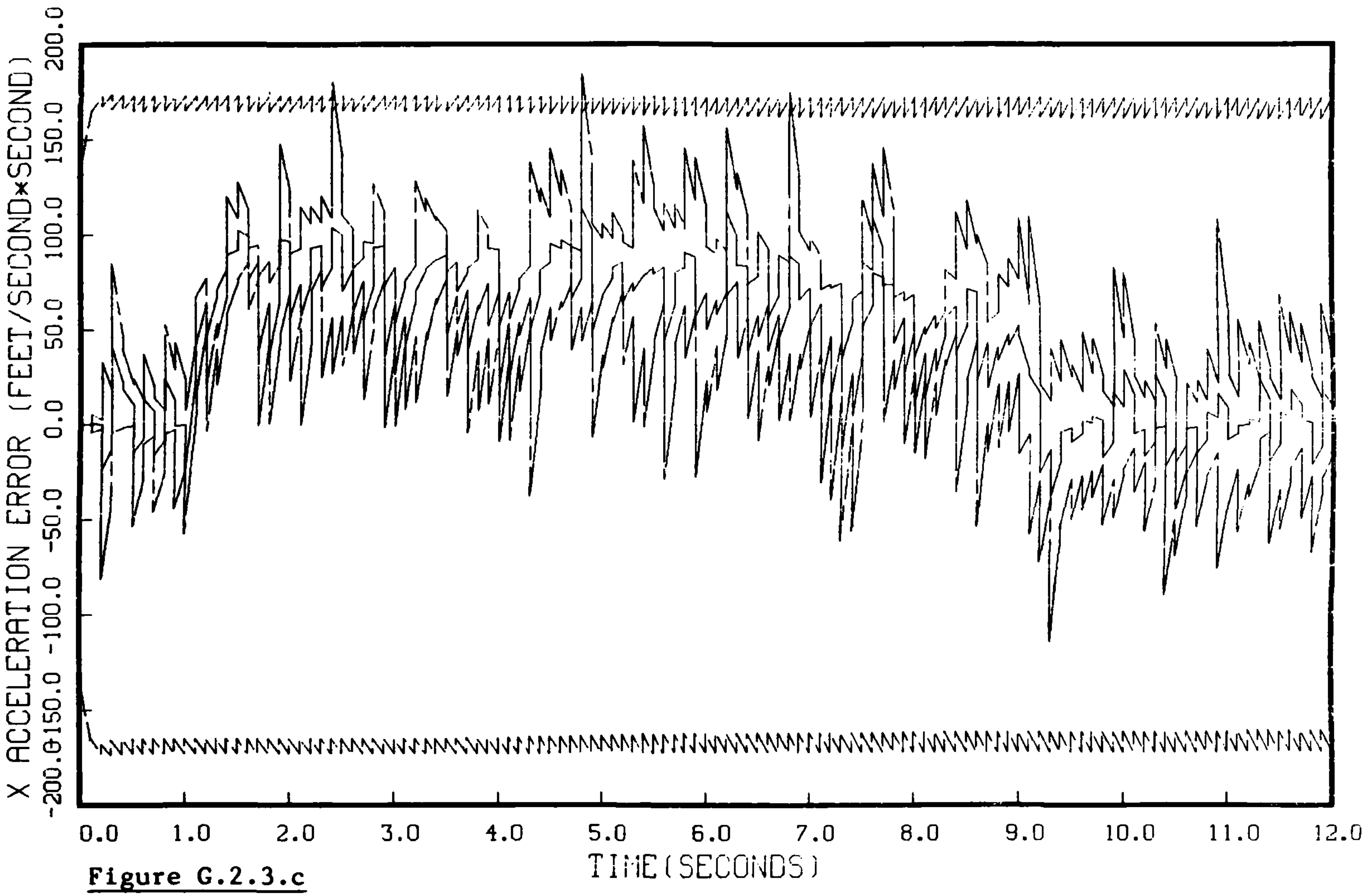


Figure G.2.3.c

STATE 3, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.167,TAU(2-3)-.167, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

4 SOFEPL DATE AND TIME - [illegible] SOFE DATE AND TIME - 85/10/02. 21.55.17. WPAFB, DISSPLA, SOFEPL VERSION 2.2

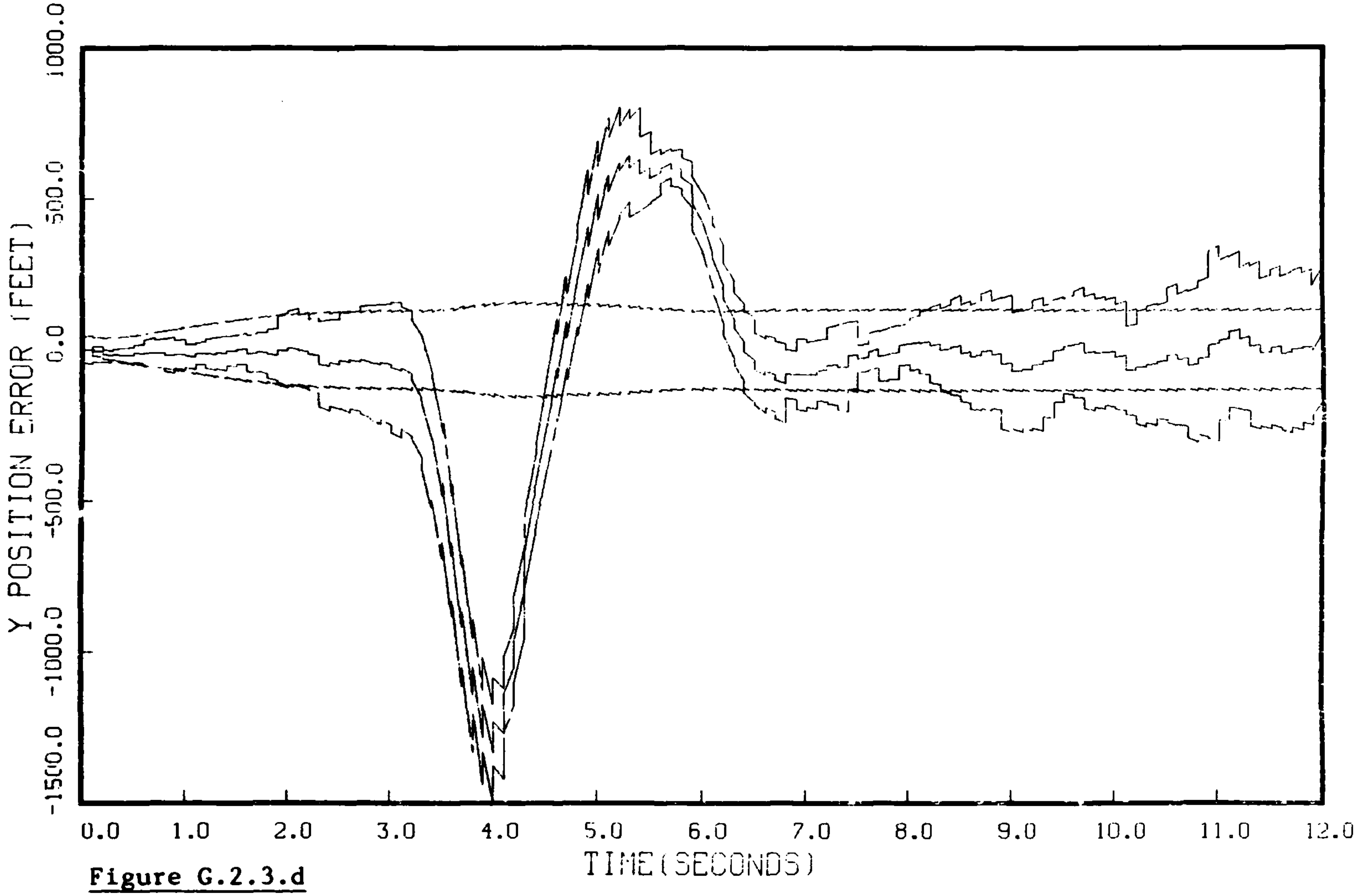


Figure G.2.3.d

STATE 4, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.167,TAU(2-3)=.167, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

5 SOFEPL DATE AND TIME - 85/10/02. 22.33.01. SOFE DATE AND TIME - 85/10/02. 21.55.17. WPAFB, DISSPLA, SOFEPL VERSION 2.2

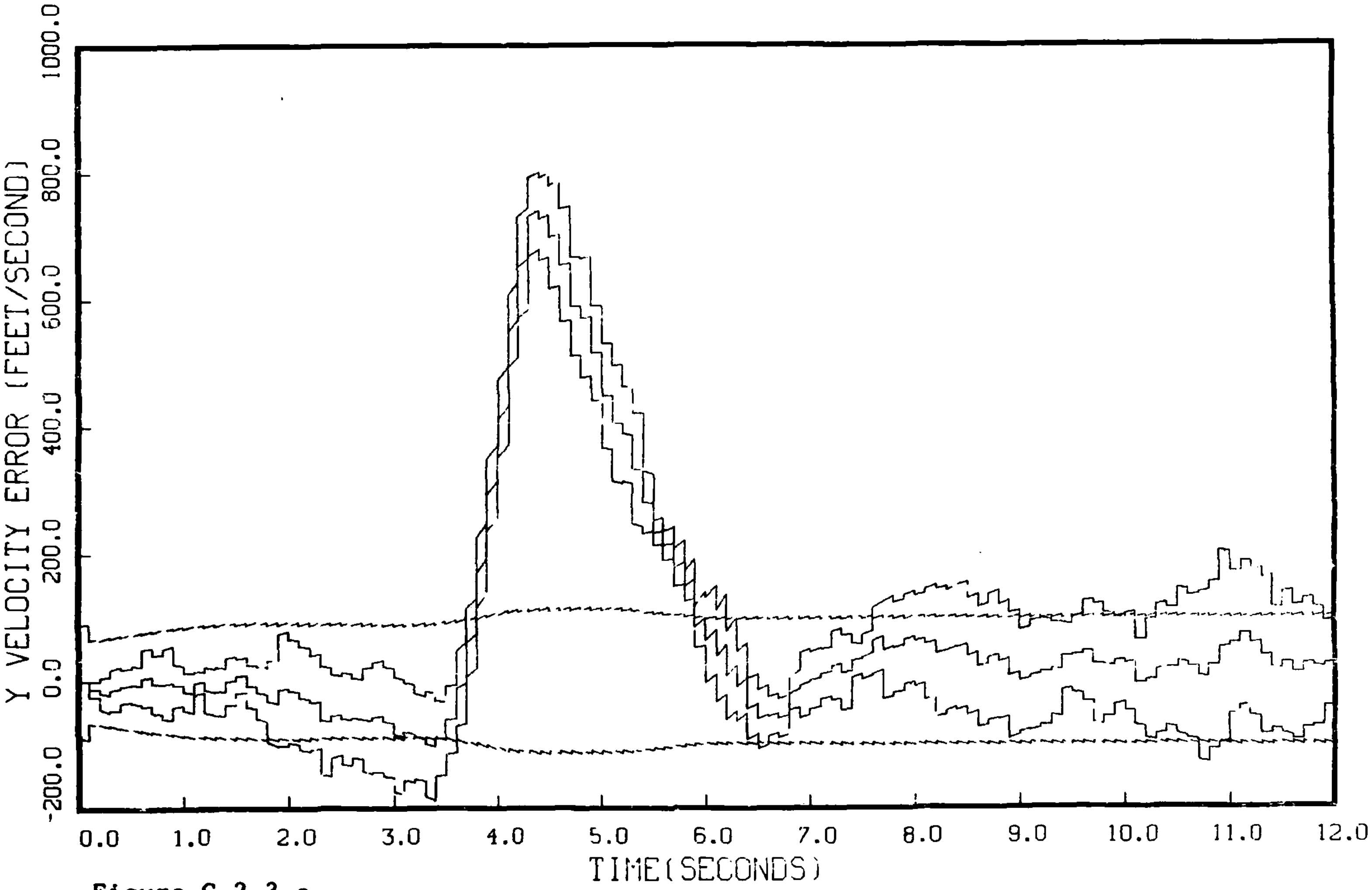


Figure G.2.3.e
STATE 5, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.167,TAU(2-3)-.167, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

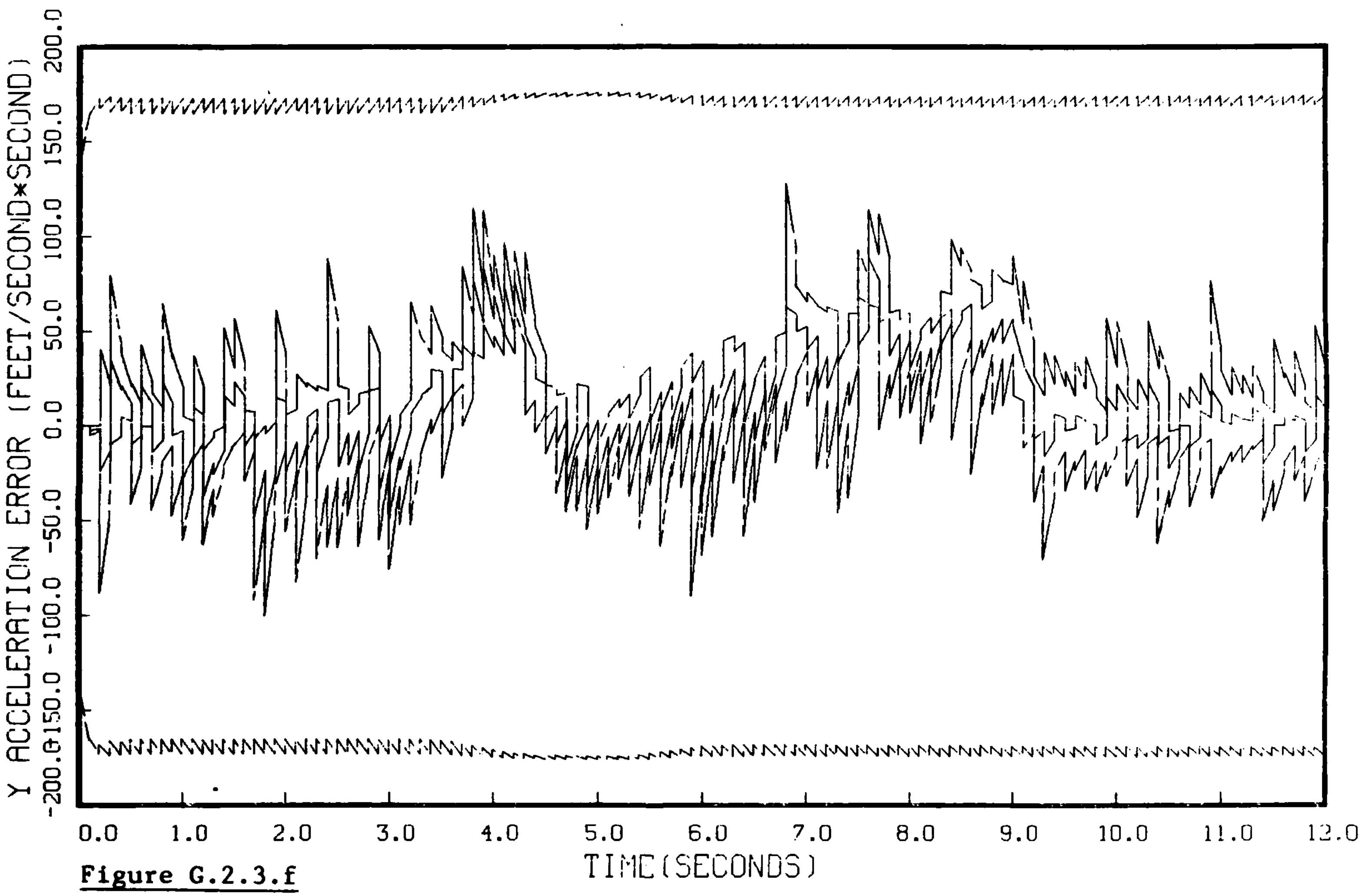


Figure G.2.3.f

STATE 6, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.167,TAU(2-3)=.167, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

7 SOFEPL DATE AND TIME - 85/10/02. 22.73.01. SOFE DATE AND TIME - 85/10/02. 21.55.17. WPAFB, DISSPLA, SOFEPL VERSION 2.2

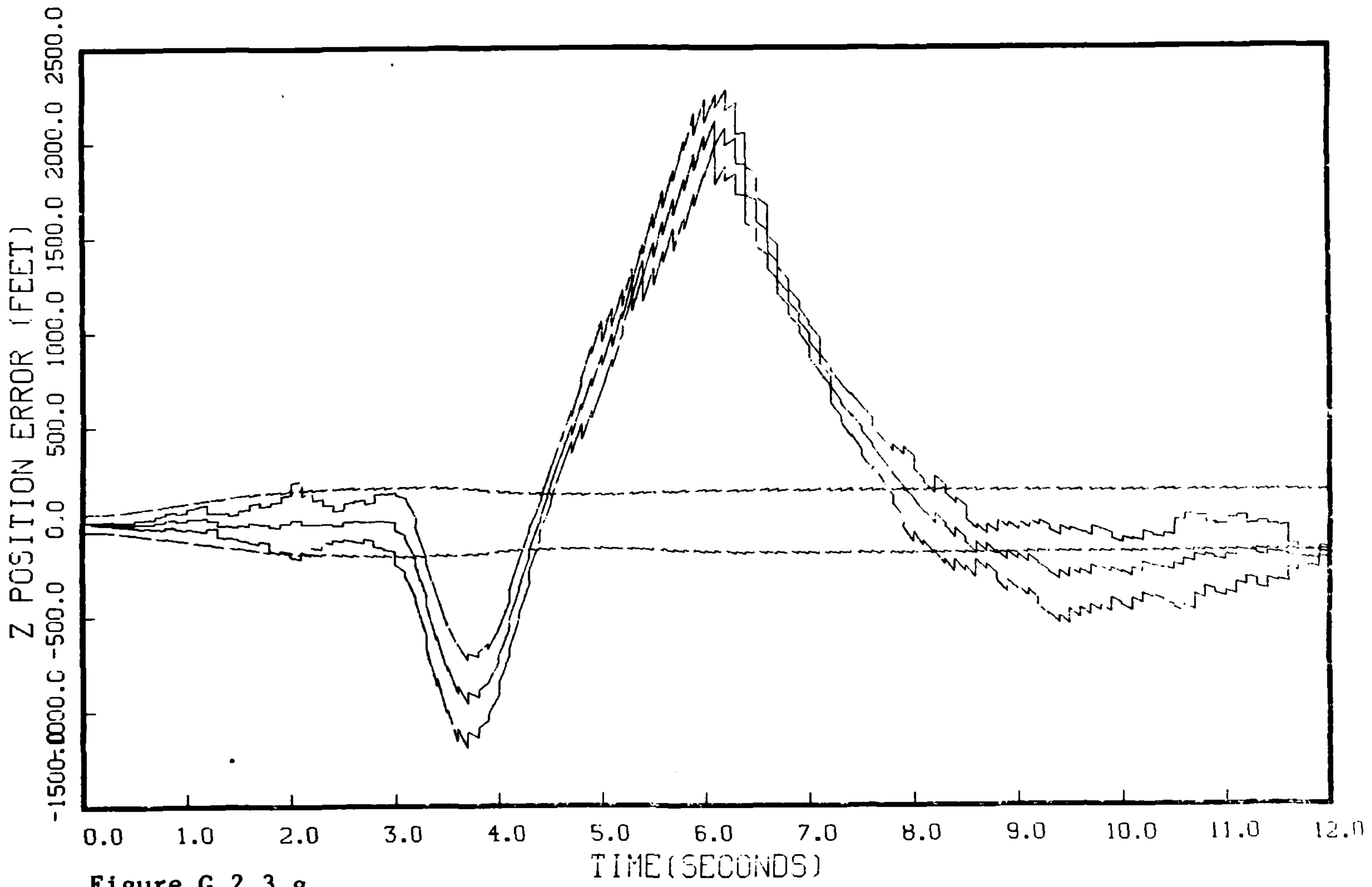


Figure G.2.3.g

STATE 7, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.167,TAU(2-3)-.167, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

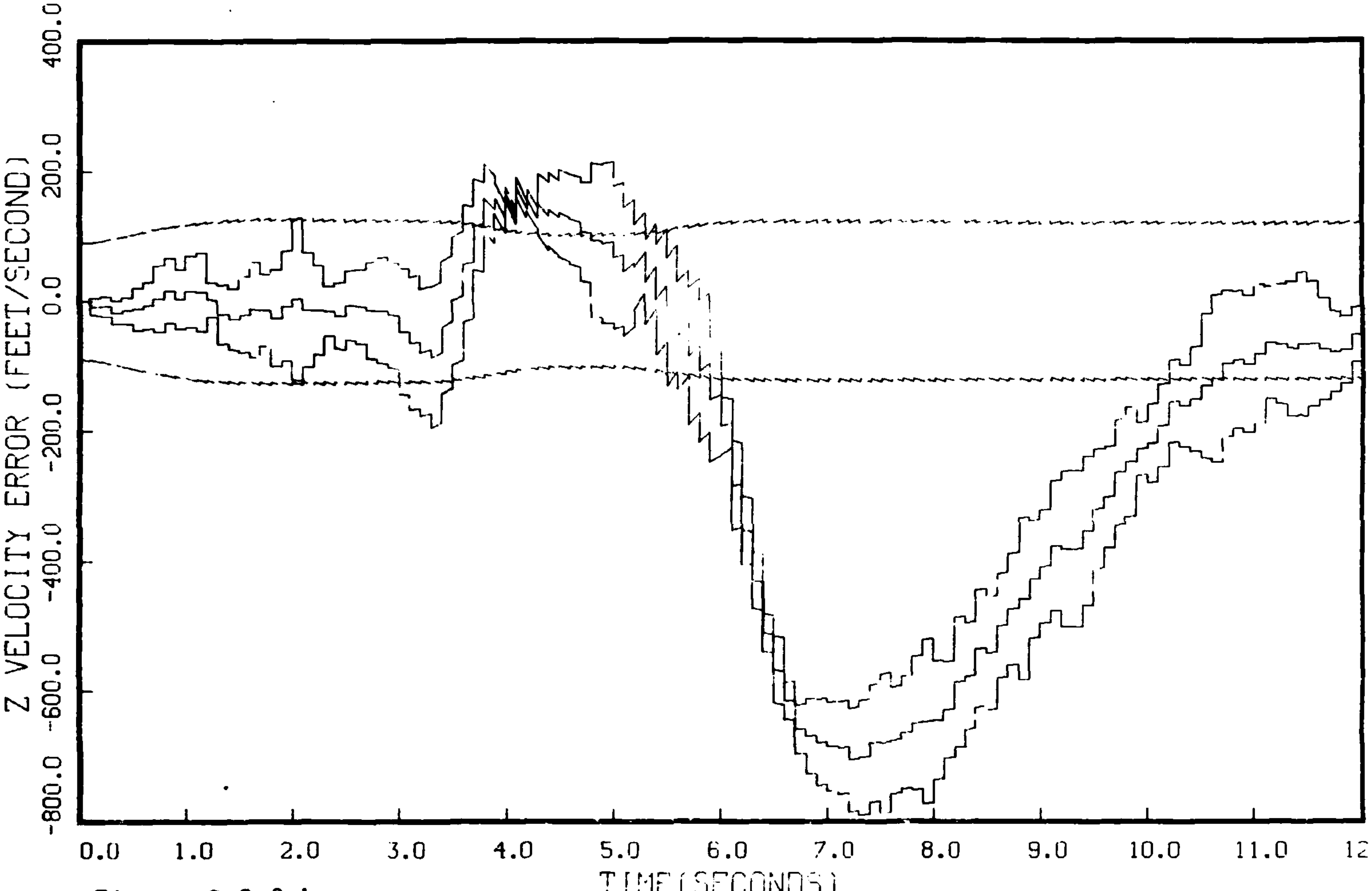


Figure G.2.3.h

STATE 8, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.167,TAU(2-3)-.167, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

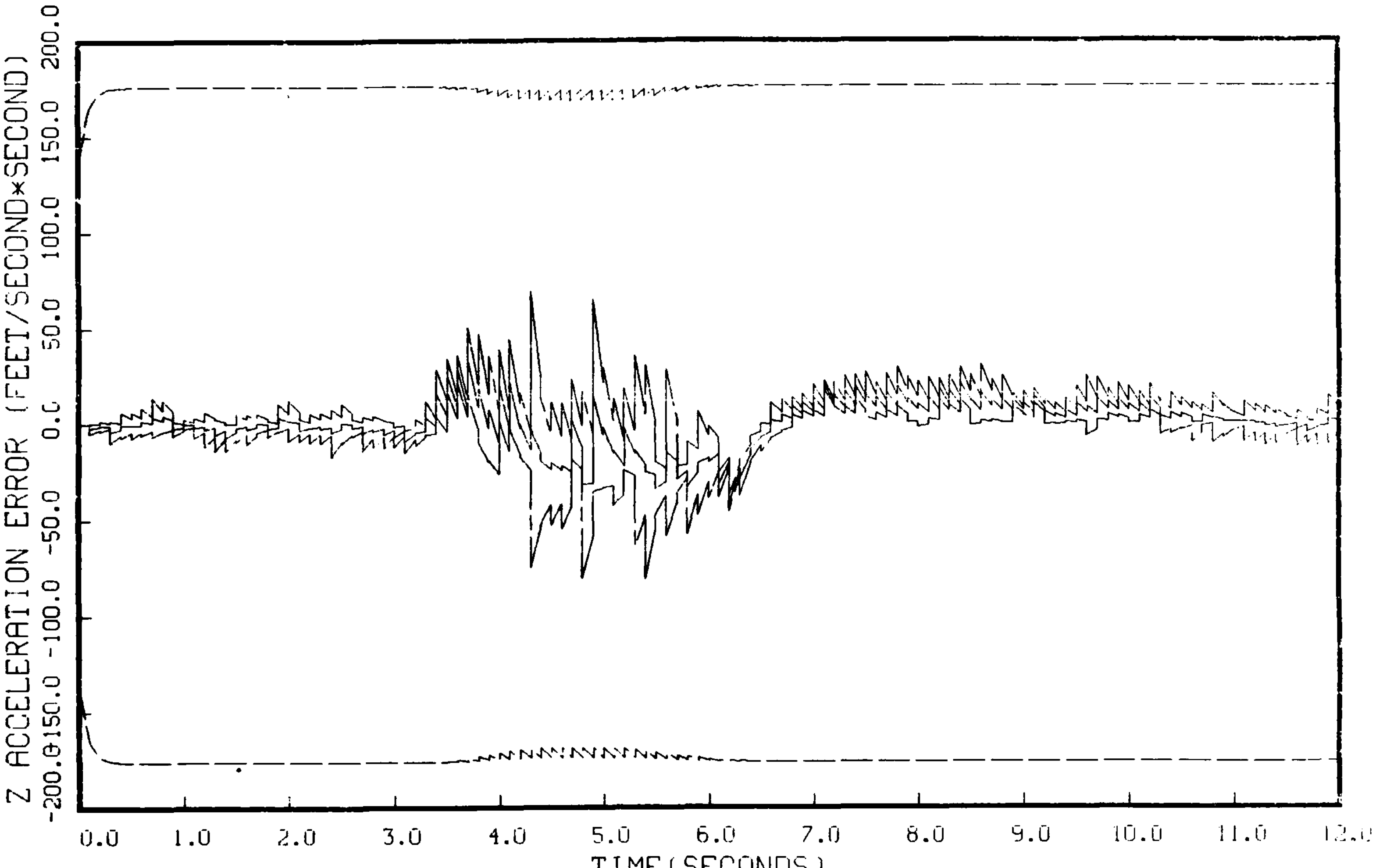


Figure G.2.3.i

STATE 9, O(1)-O(2)-O(3)-373250., TAU(1)-.167,TAU(2-3)-.167, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

64 PS2 4 1 SOFEPL DATE AND TIME - 85/10/02. 15.36.10. SOFE DATE AND TIME - 85/10/02. 14.34.52. WPAFE, DISSPLA, SOFEPL VERSION 2.2

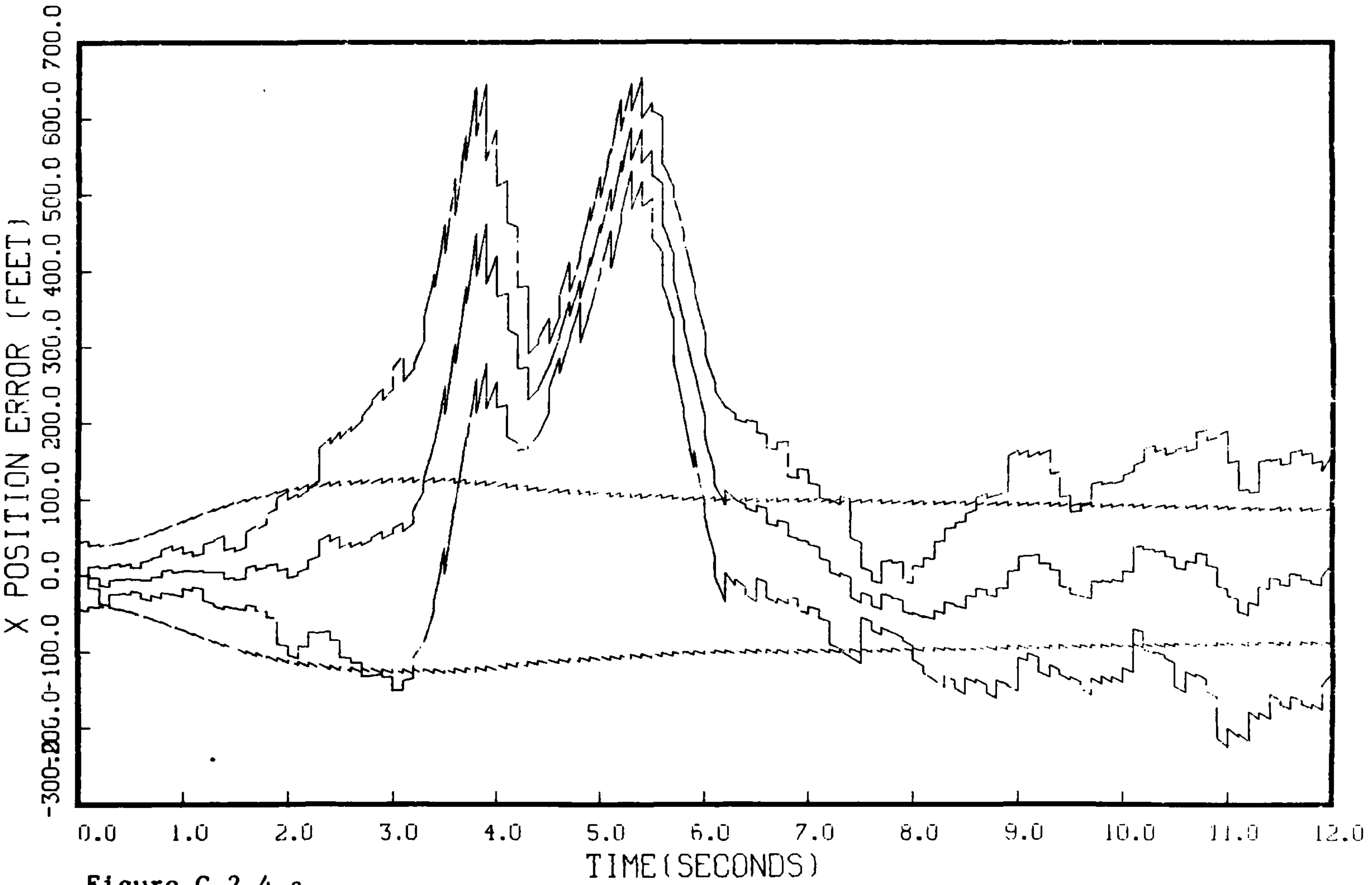


Figure G.2.4.a

STATE 1, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

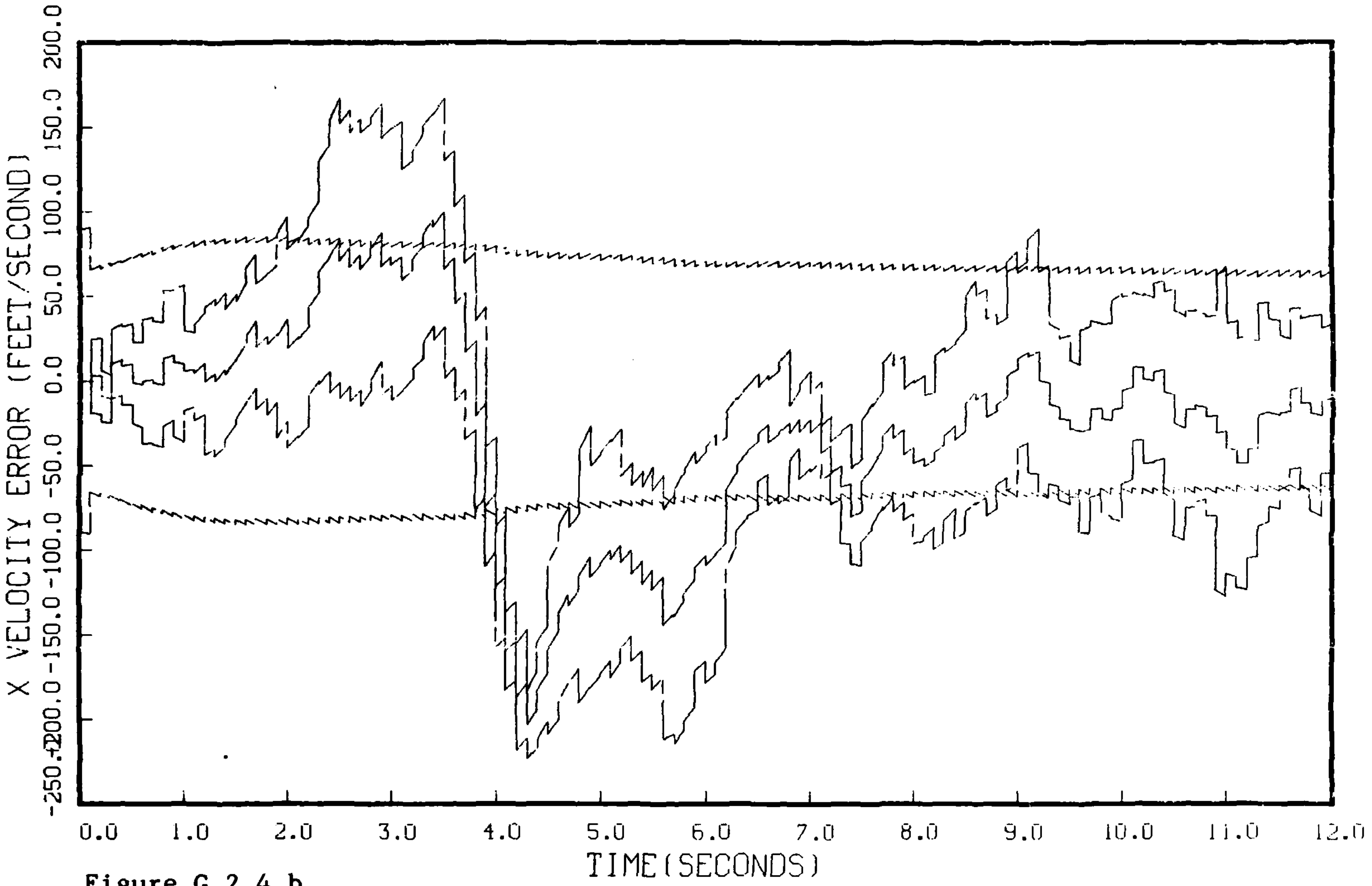


Figure G.2.4.b

STATE 2, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

3 SOFEPL DATE AND TIME - 85/10/02. 15.36.10. SOFE DATE AND TIME - 85/10/02. 14.34.52. WPAFB, DISSPLA, SOFEPL VERSION 2.2

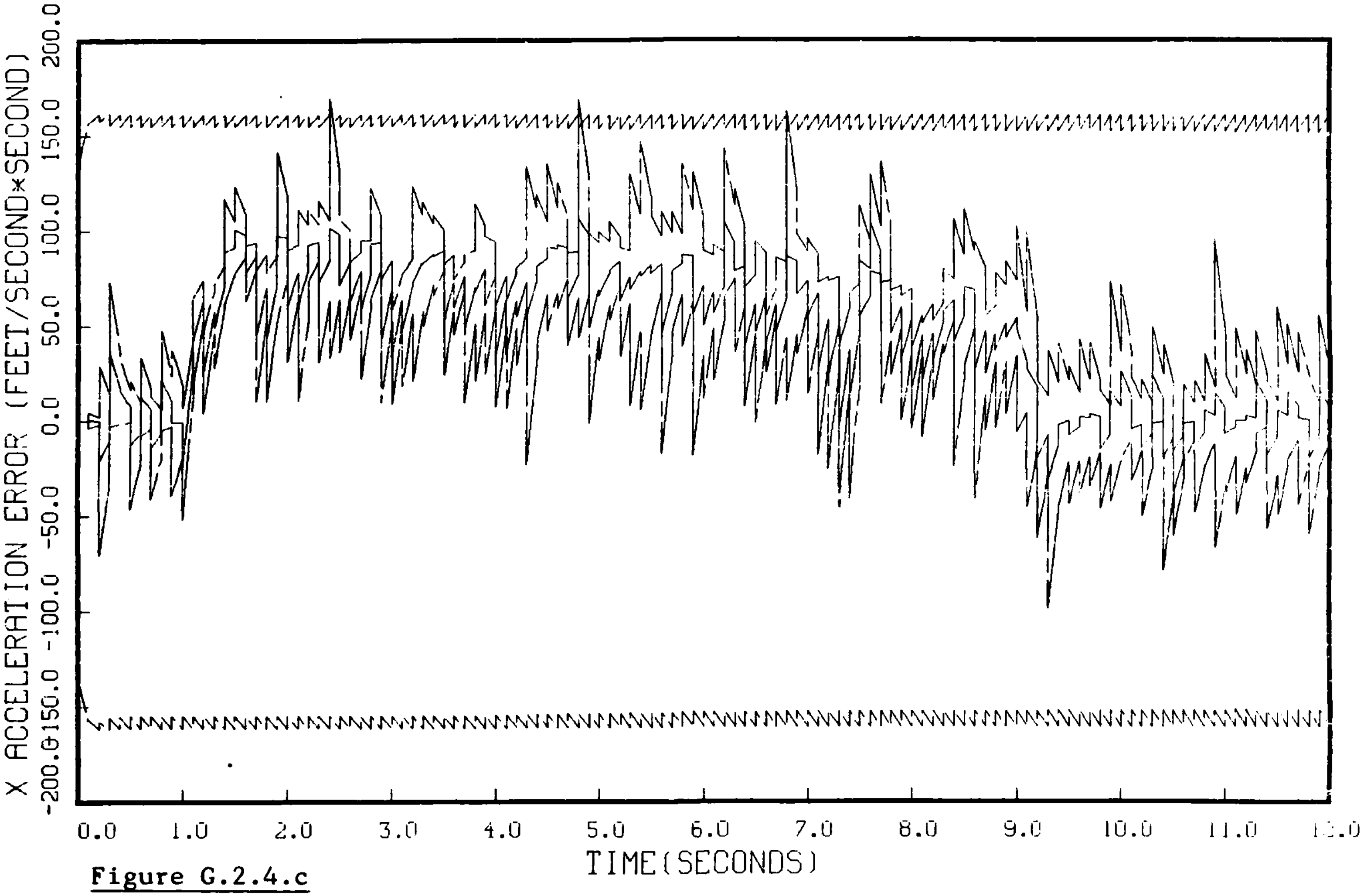


Figure G.2.4.c

STATE 3, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

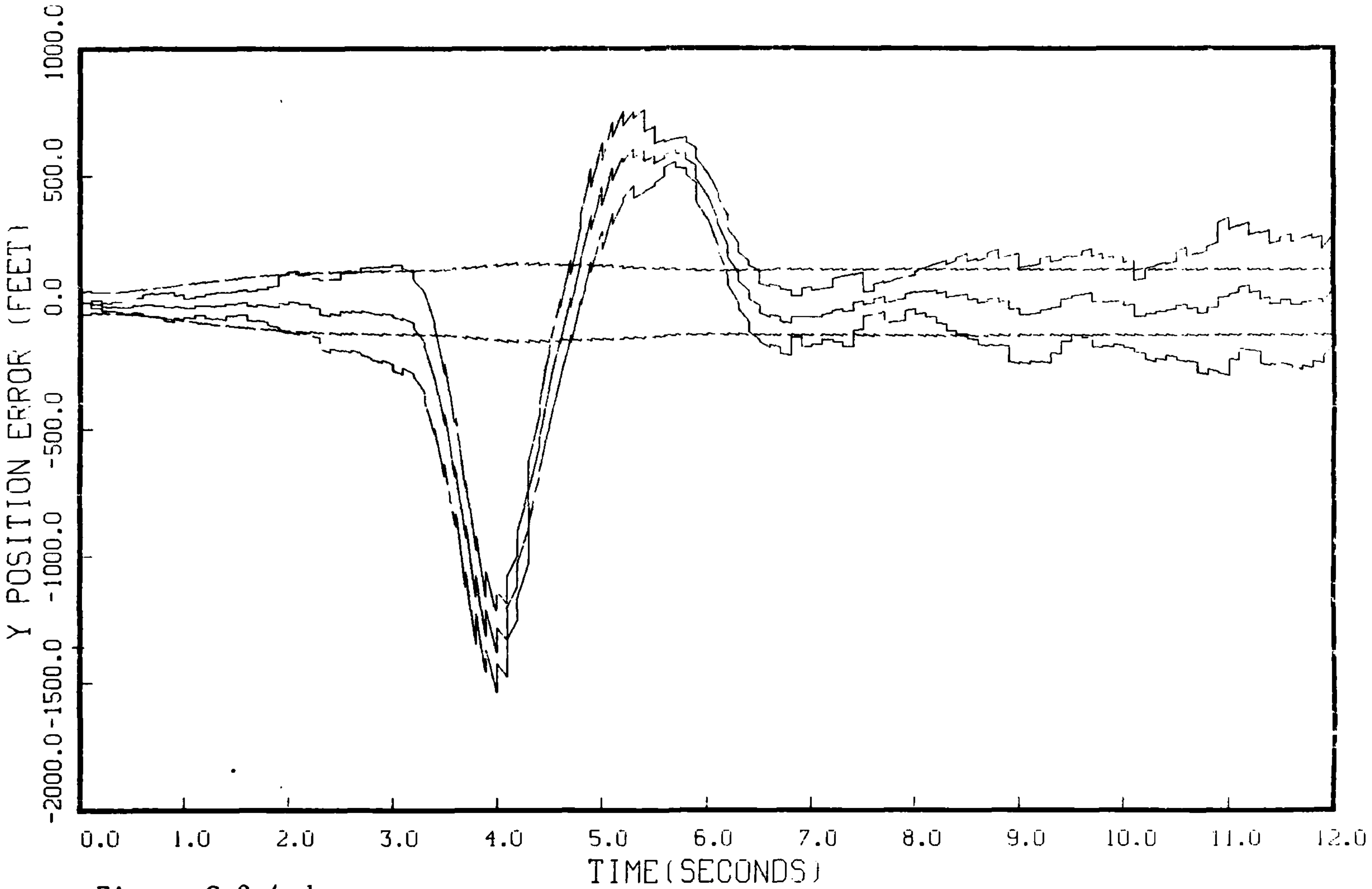


Figure G.2.4.d
STATE 4, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

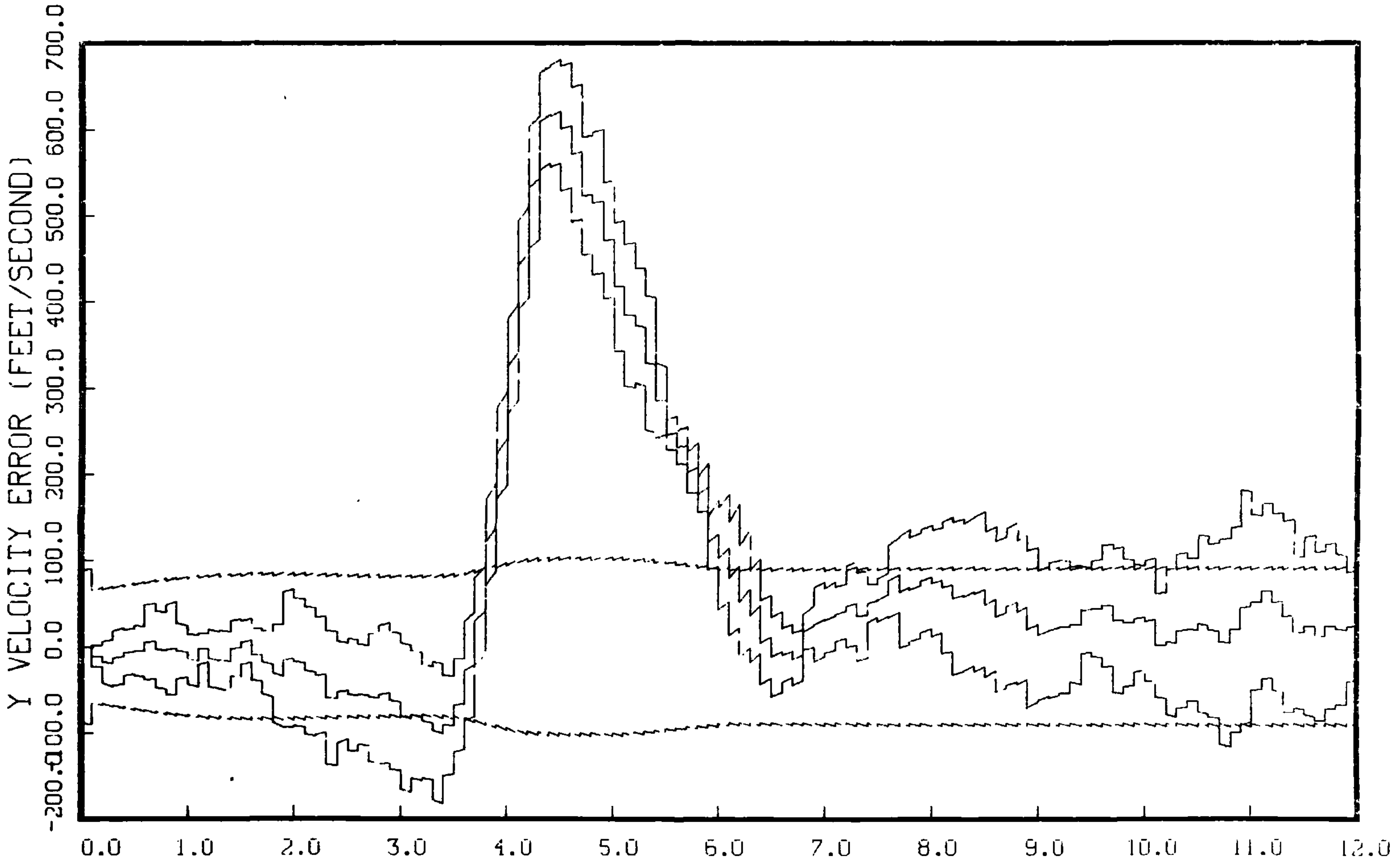


Figure G.2.4.e

STATE 5, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

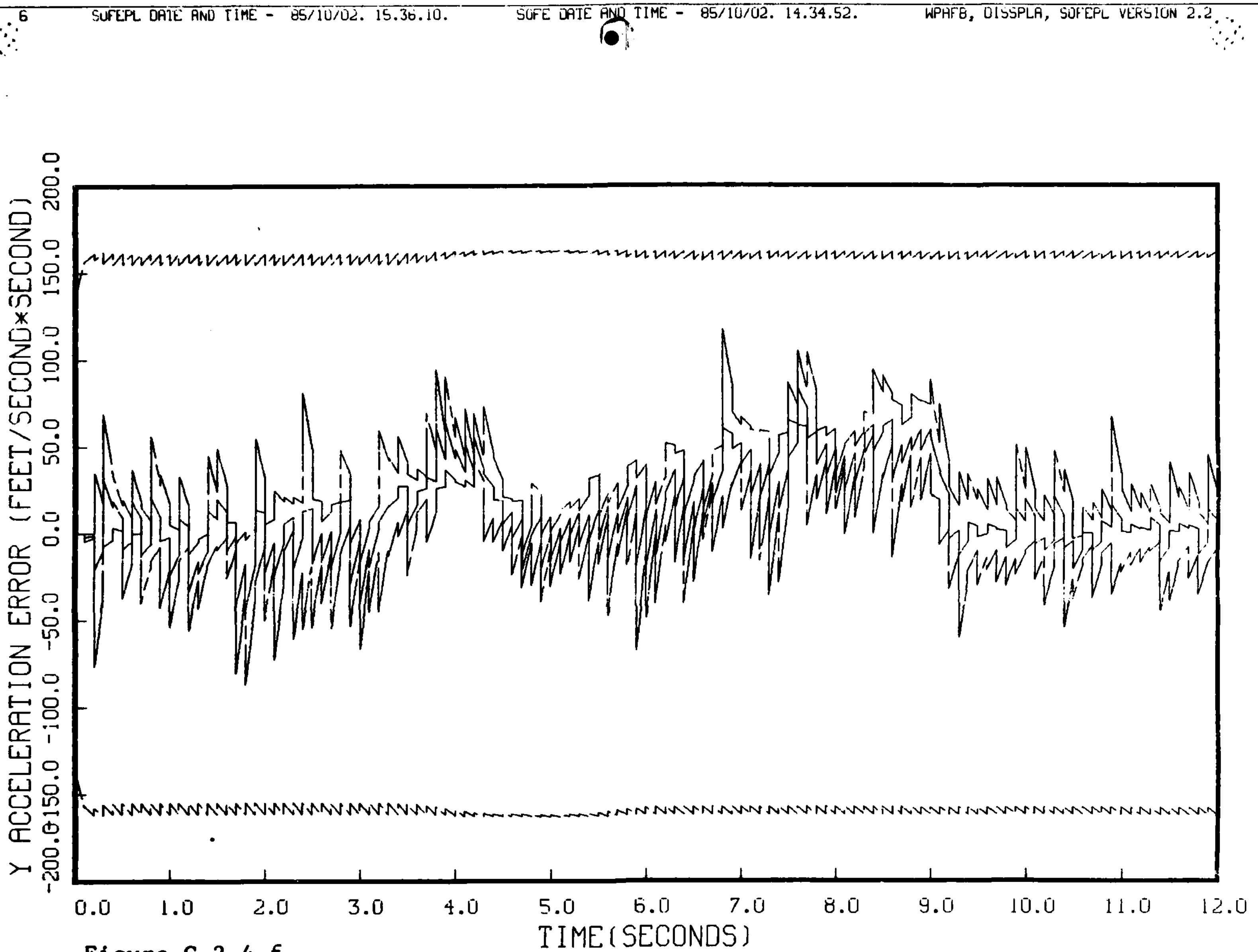


Figure G.2.4.f

STATE 6, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

SOFEPL DATE AND TIME - 85/10/02. 15.36.10. SOFE DATE AND TIME - 85/10/02. 14.34.52. WPAFB, DISSPLA, SOFEPL VERSION 2.2

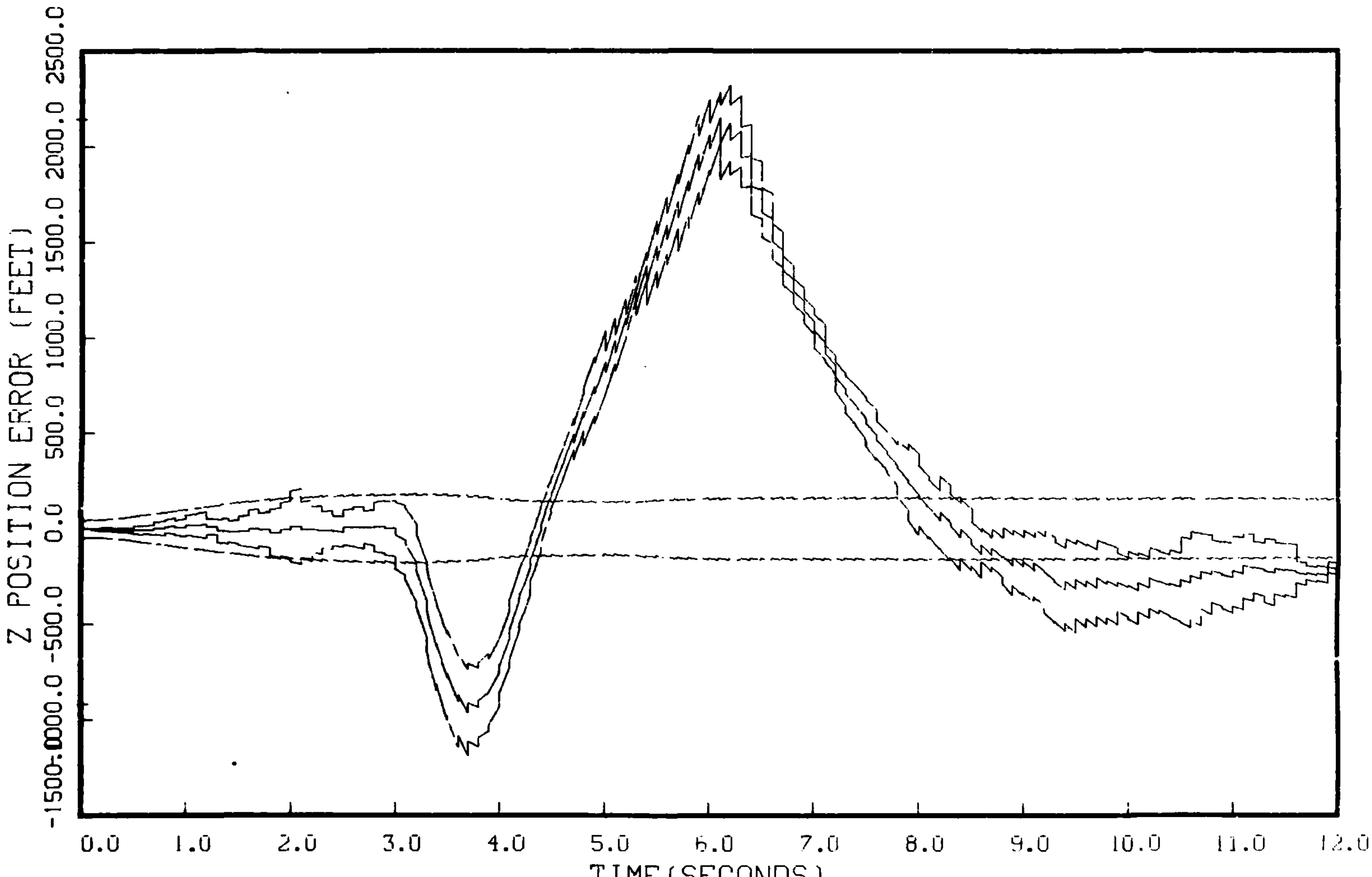


Figure G.2.4.g

STATE 7, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

8 SOFEPL DATE AND TIME - 85/10/02. 15.36.10. SOFE DATE AND TIME - 85/10/02. 14.34.52. WPAFB, DISSPLA, SOFEPL VERSION 2.2

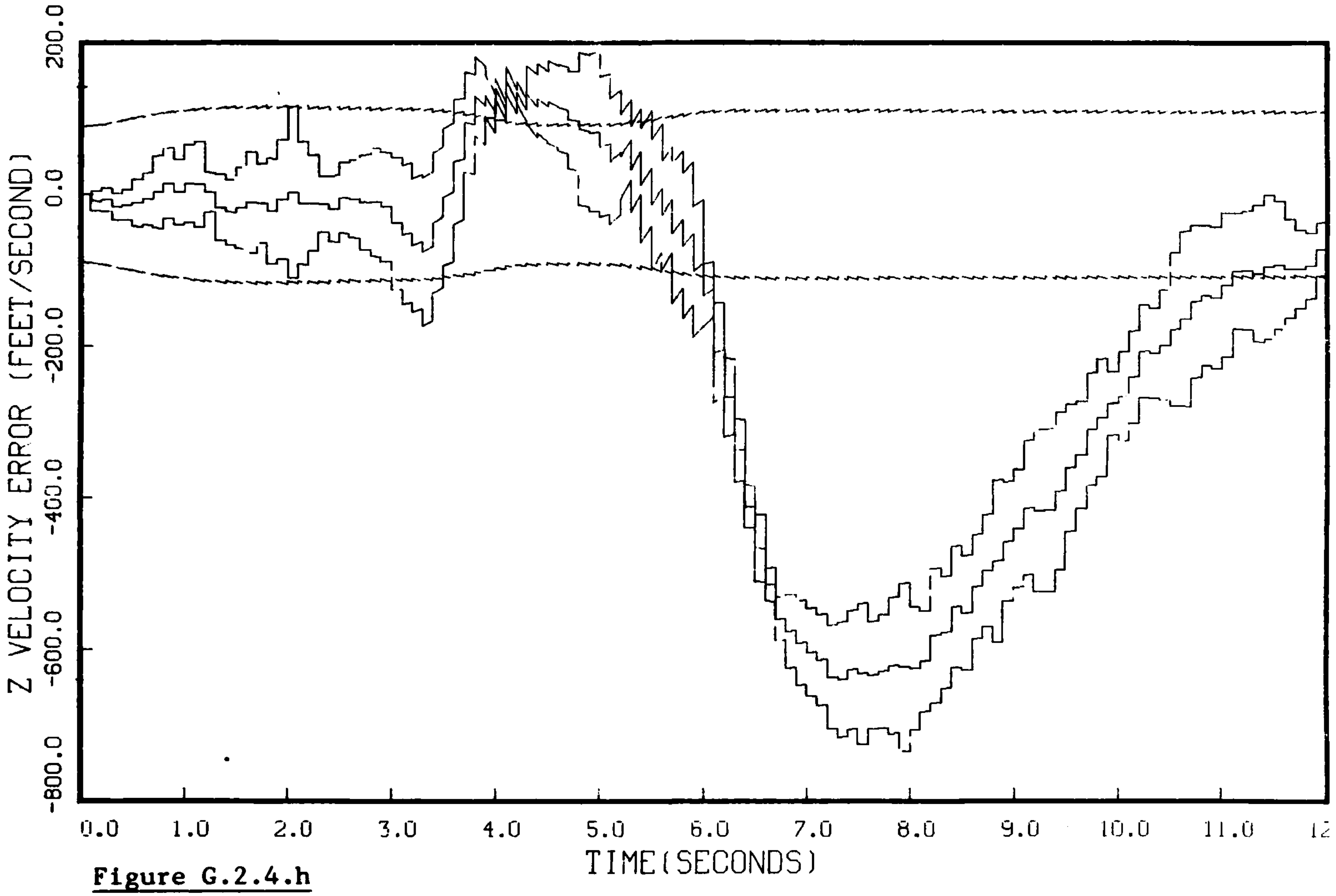


Figure G.2.4.h

STATE 8, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

9 SOFEPL DATE AND TIME - 85/10/02. 15.36.10. SOFE DATE AND TIME - 85/10/02. 14.34.52. WPAFB, DISSPLA, SOFEPL VERSION 2.2

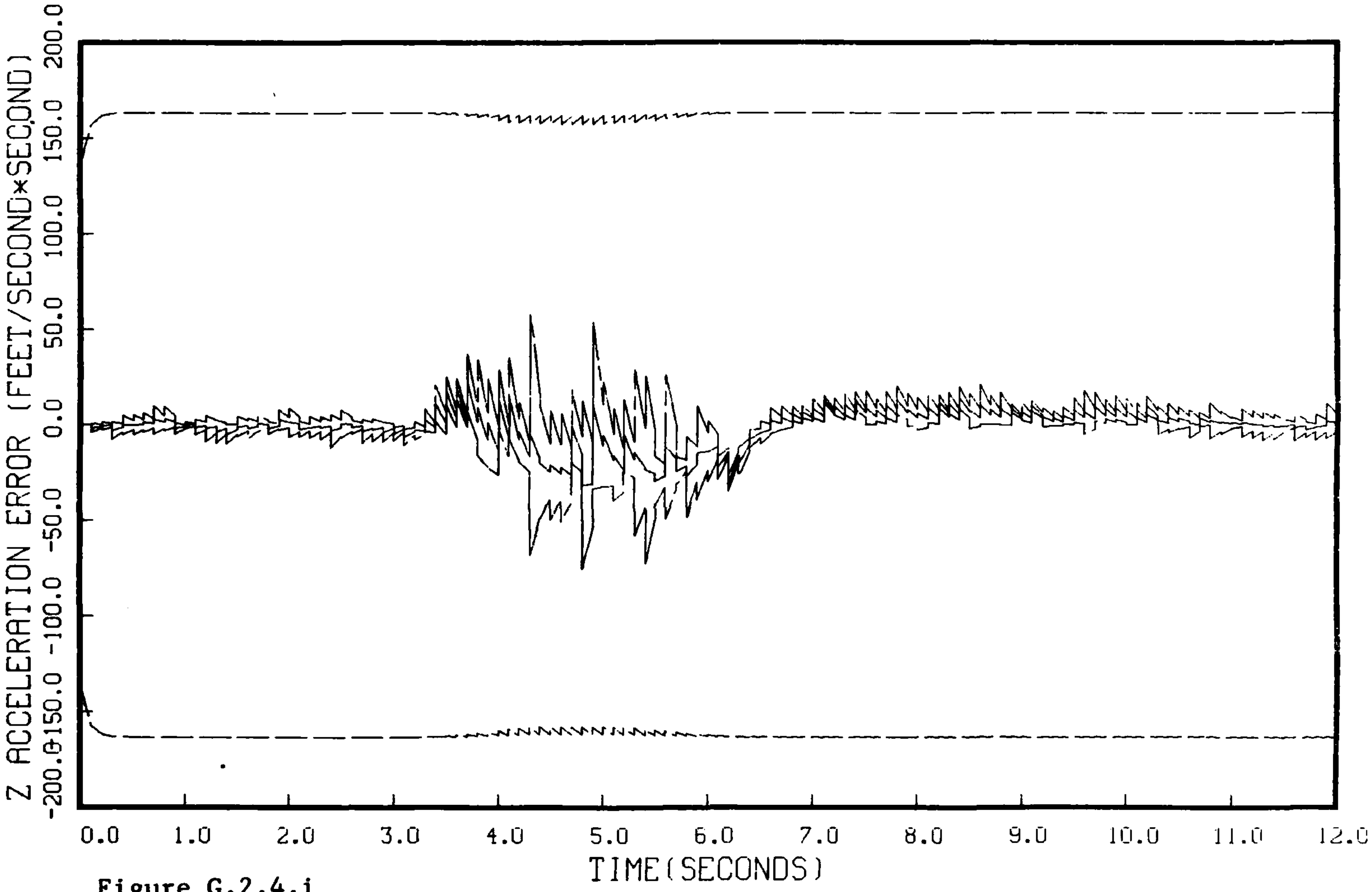


Figure G.2.4.i

STATE 9, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

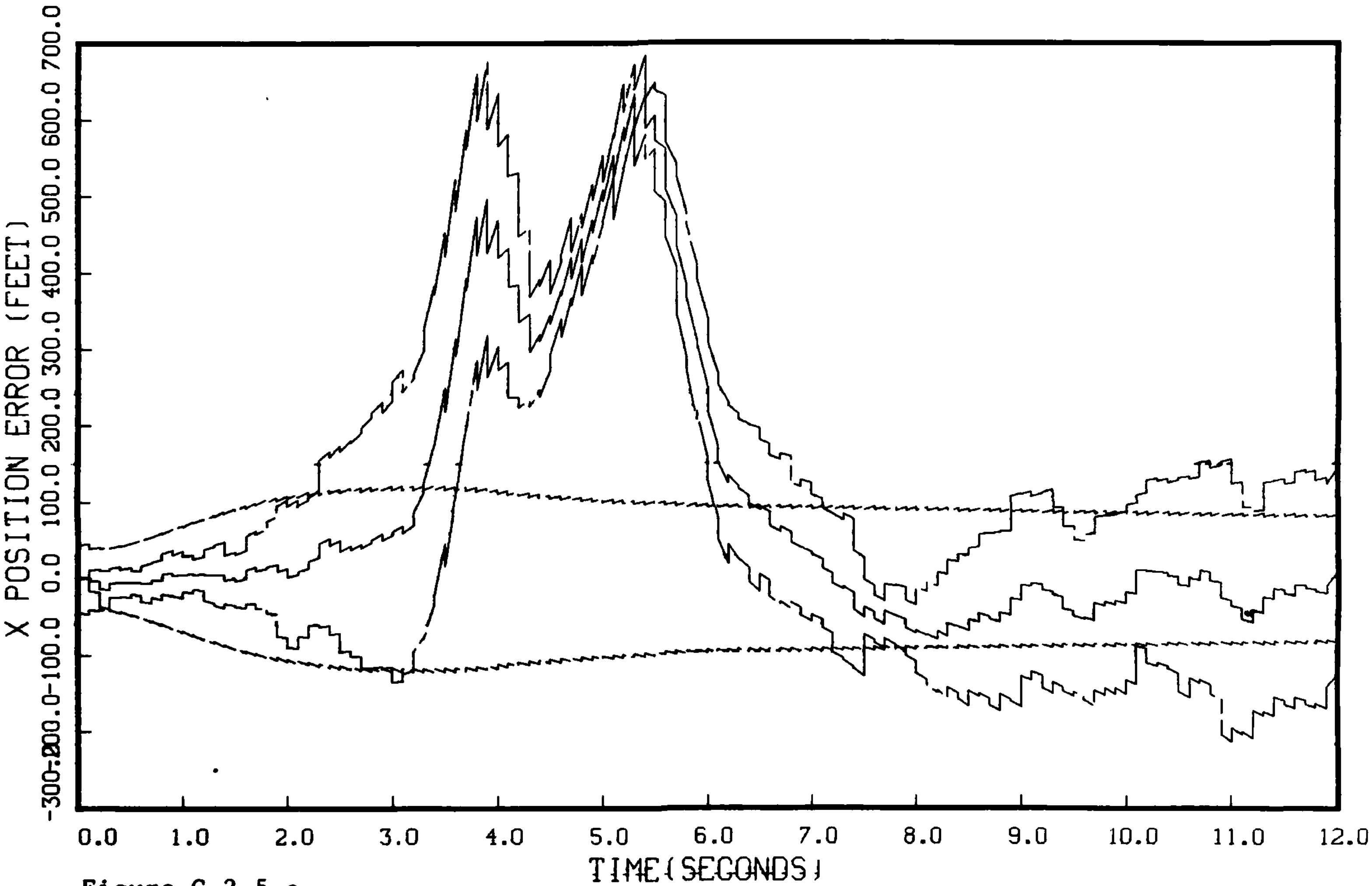


Figure G.2.5.a

STATE 1, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

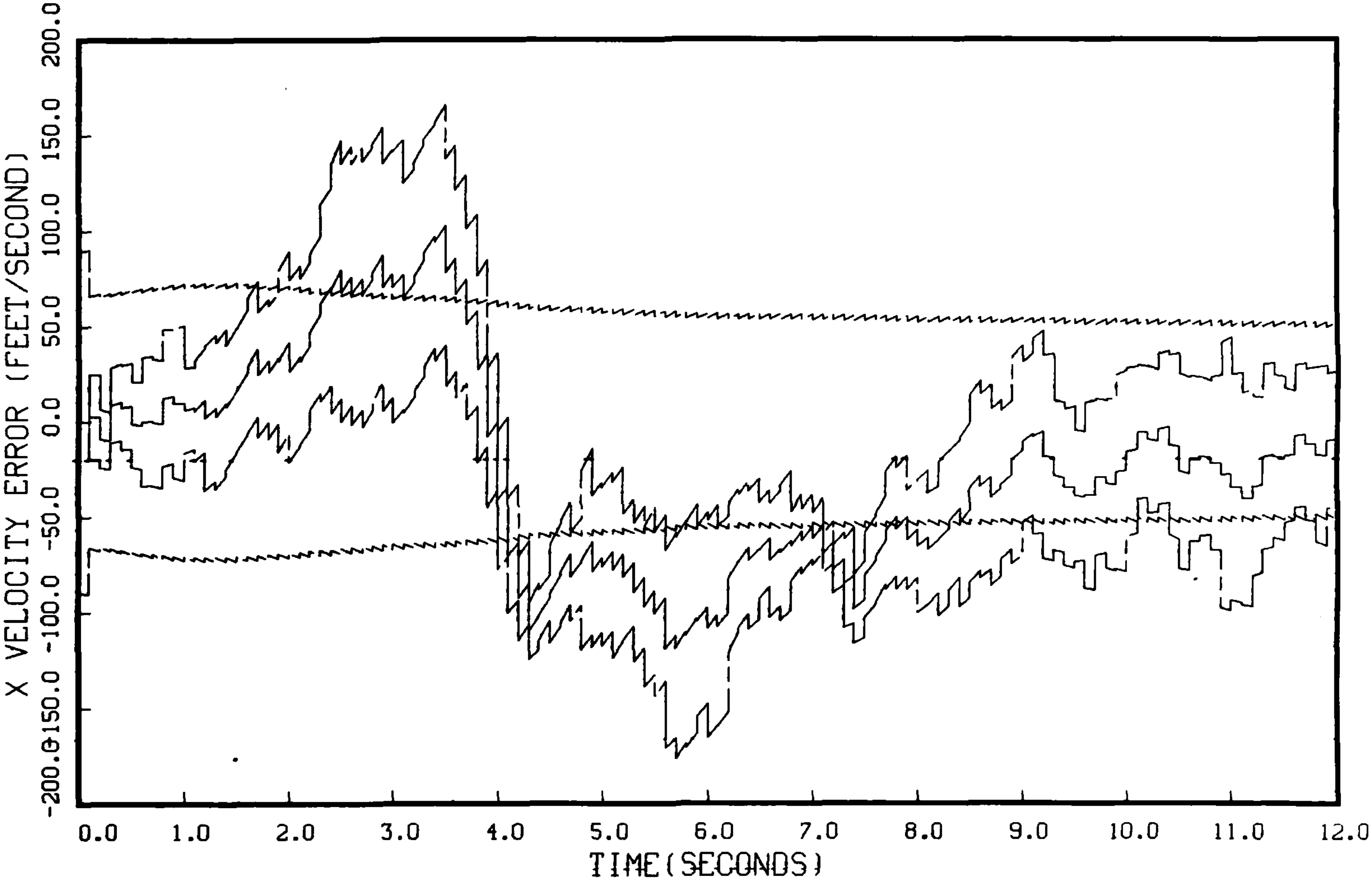


Figure G.2.5.b
STATE 2, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

3 SOFEPL DATE AND TIME - 85/10/08. 22.56.12. SOFE DATE AND TIME - 85/10/08. 22.47.35. WPAFB, DISSPLA, SOFEPL VERSION 2.2

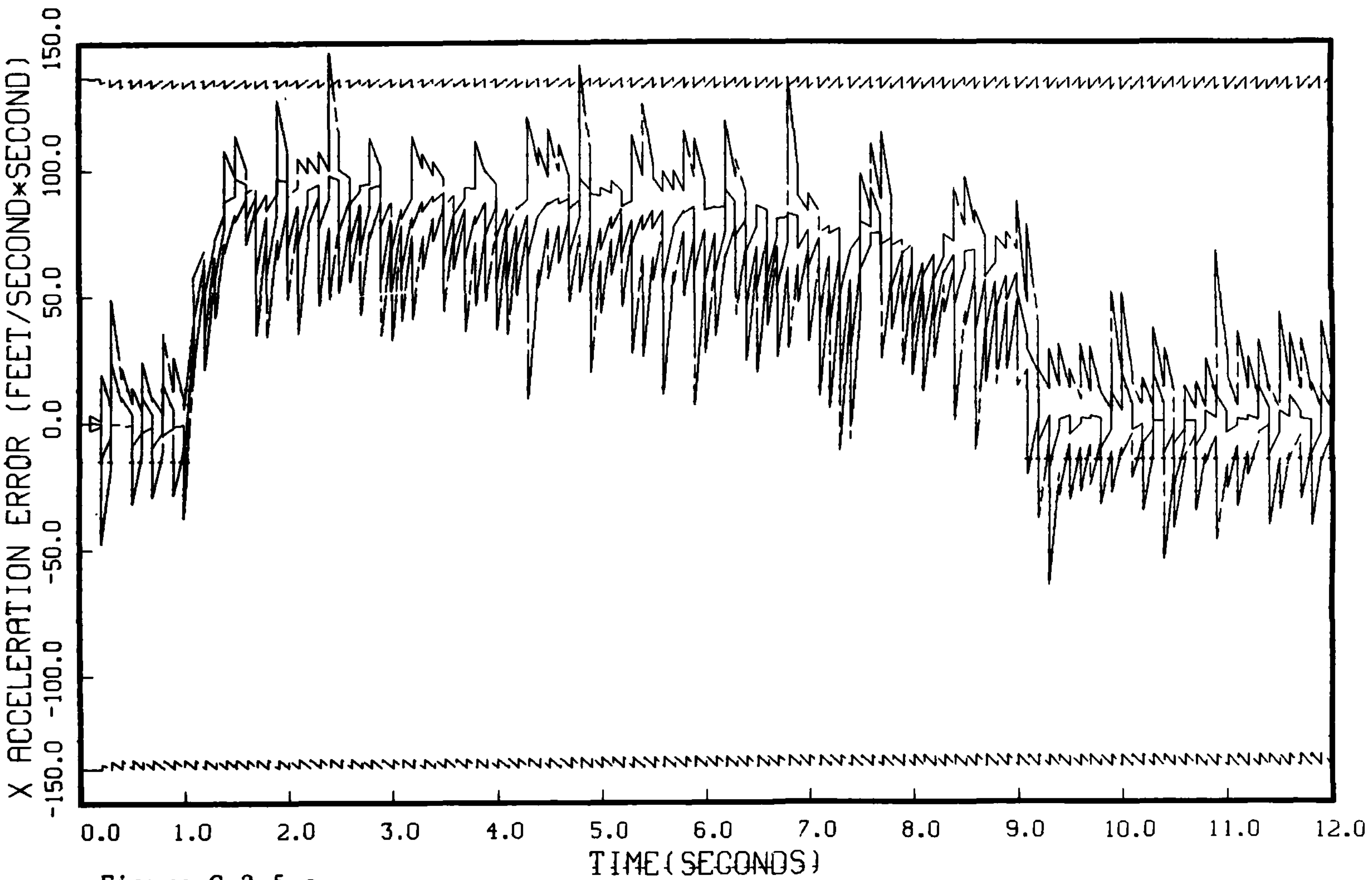


Figure G.2.5.c

STATE 3, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

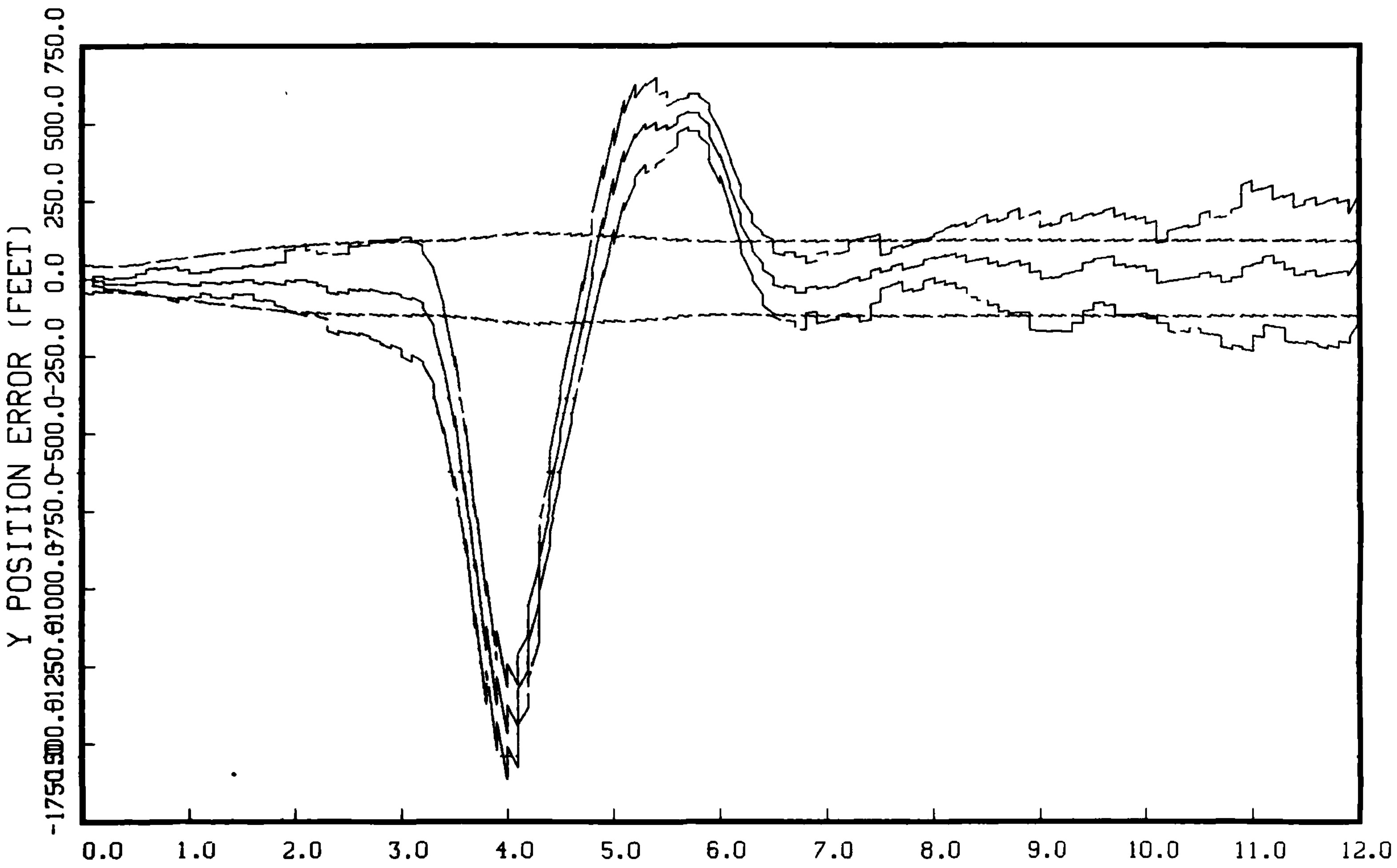


Figure G.2.5.d

STATE 4, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

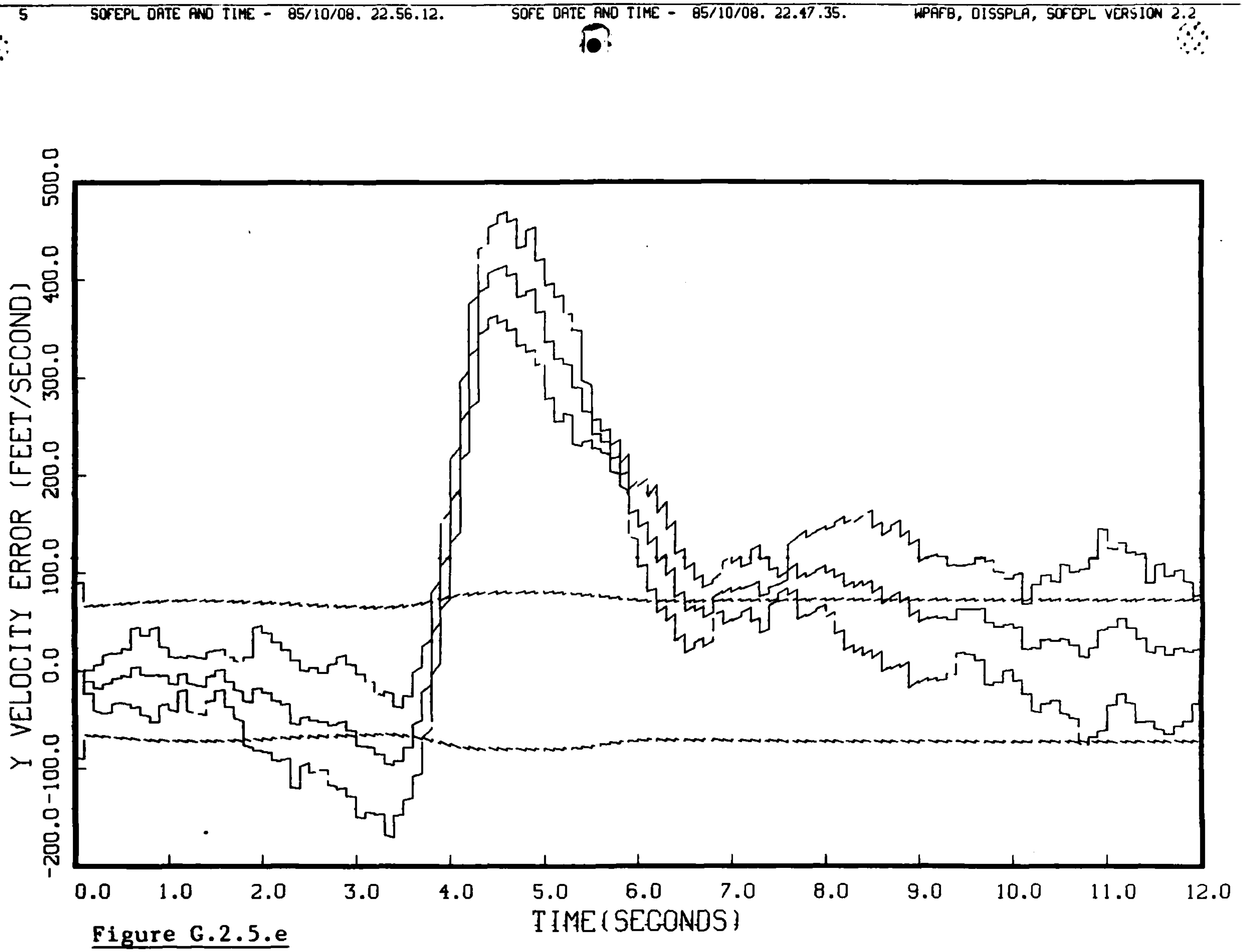


Figure G.2.5.e

STATE 5, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

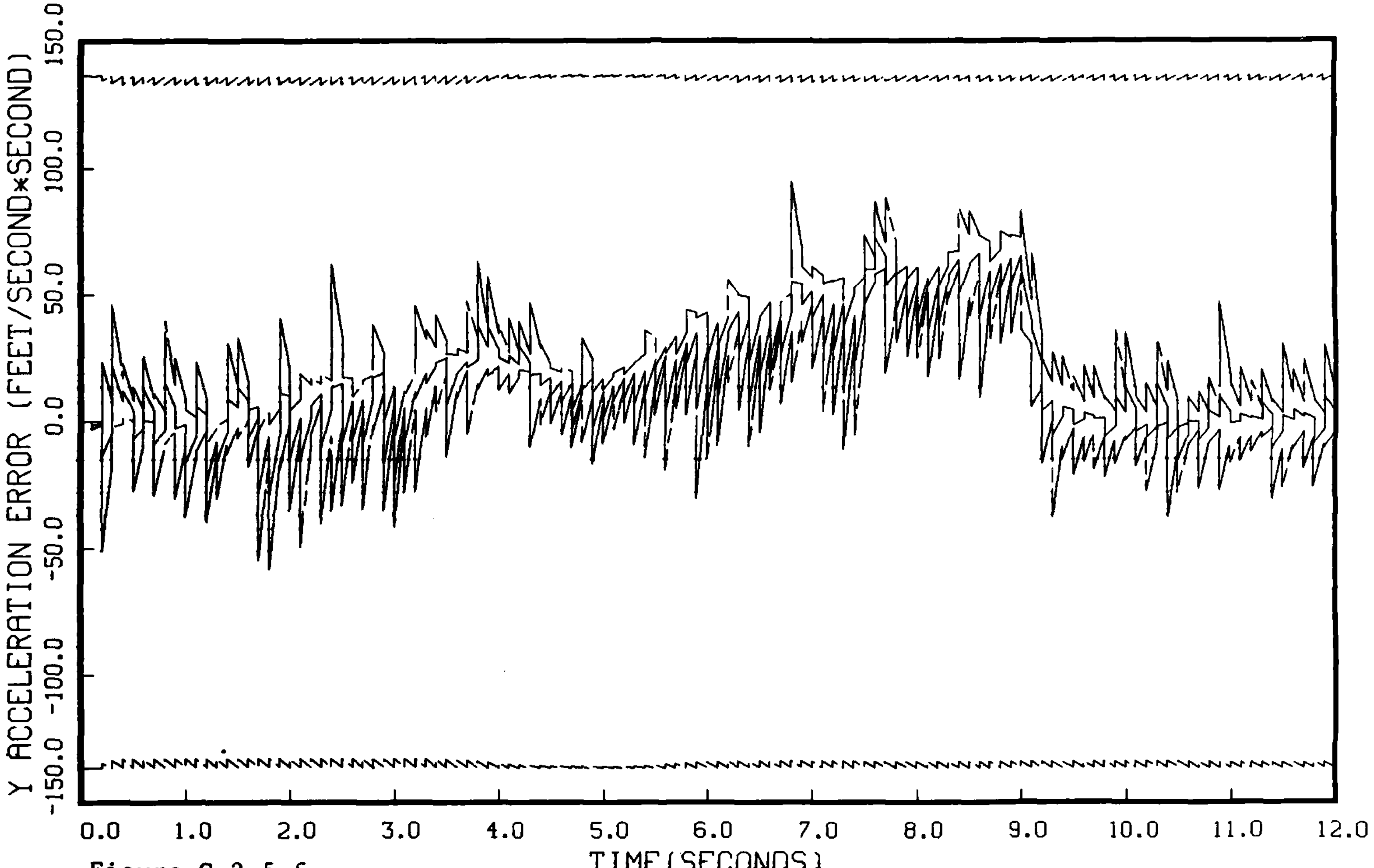


Figure G.2.5.f

STATE 6, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

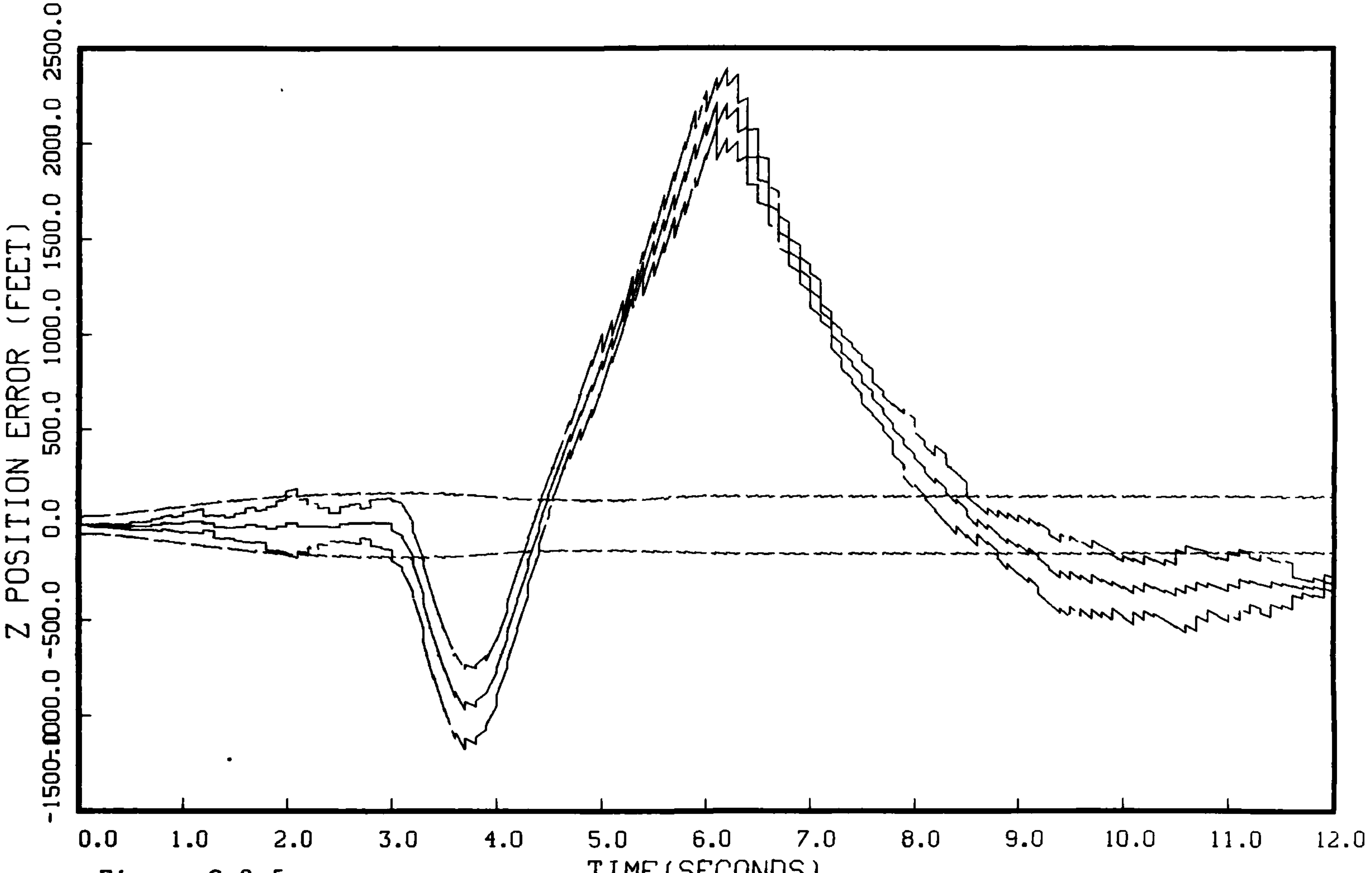


Figure G.2.5.g

STATE 7, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

8 SOFEPL DATE AND TIME - 85/10/08. 22.56.12. SOFE DATE AND TIME - 85/10/08. 22.47.35. WPAFB, DISSPLA, SOFEPL VERSION 2.?

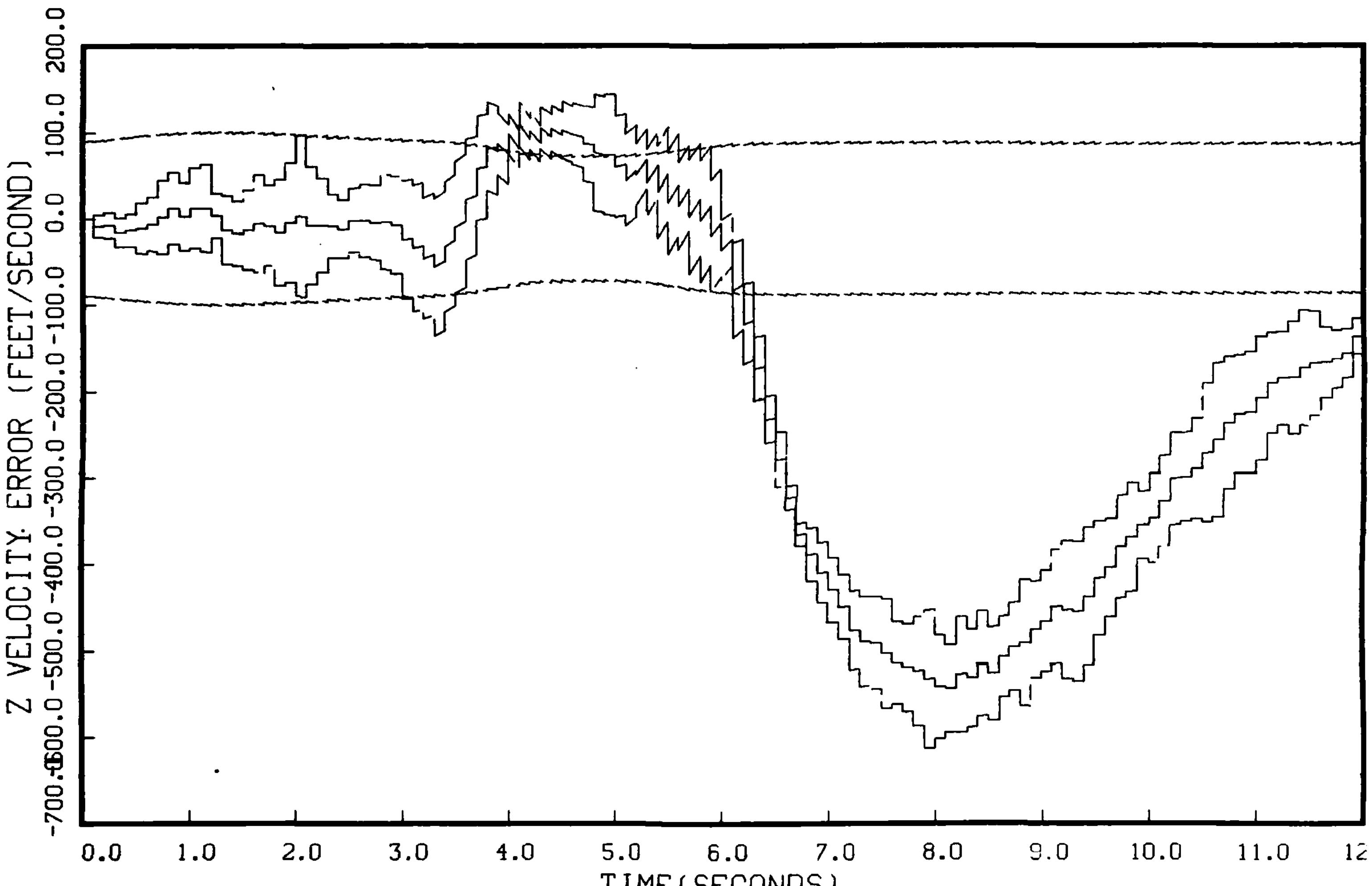


Figure G.2.5.h

STATE 8, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.10,TAU(2-3)-.10, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

9 SOFEPL DATE AND TIME - 85/10/08. 22.56.12. SOFE DATE AND TIME - 85/10/08. 22.47.35. WPAFB, DISSPLA, SOFEPL VERSION 2.2

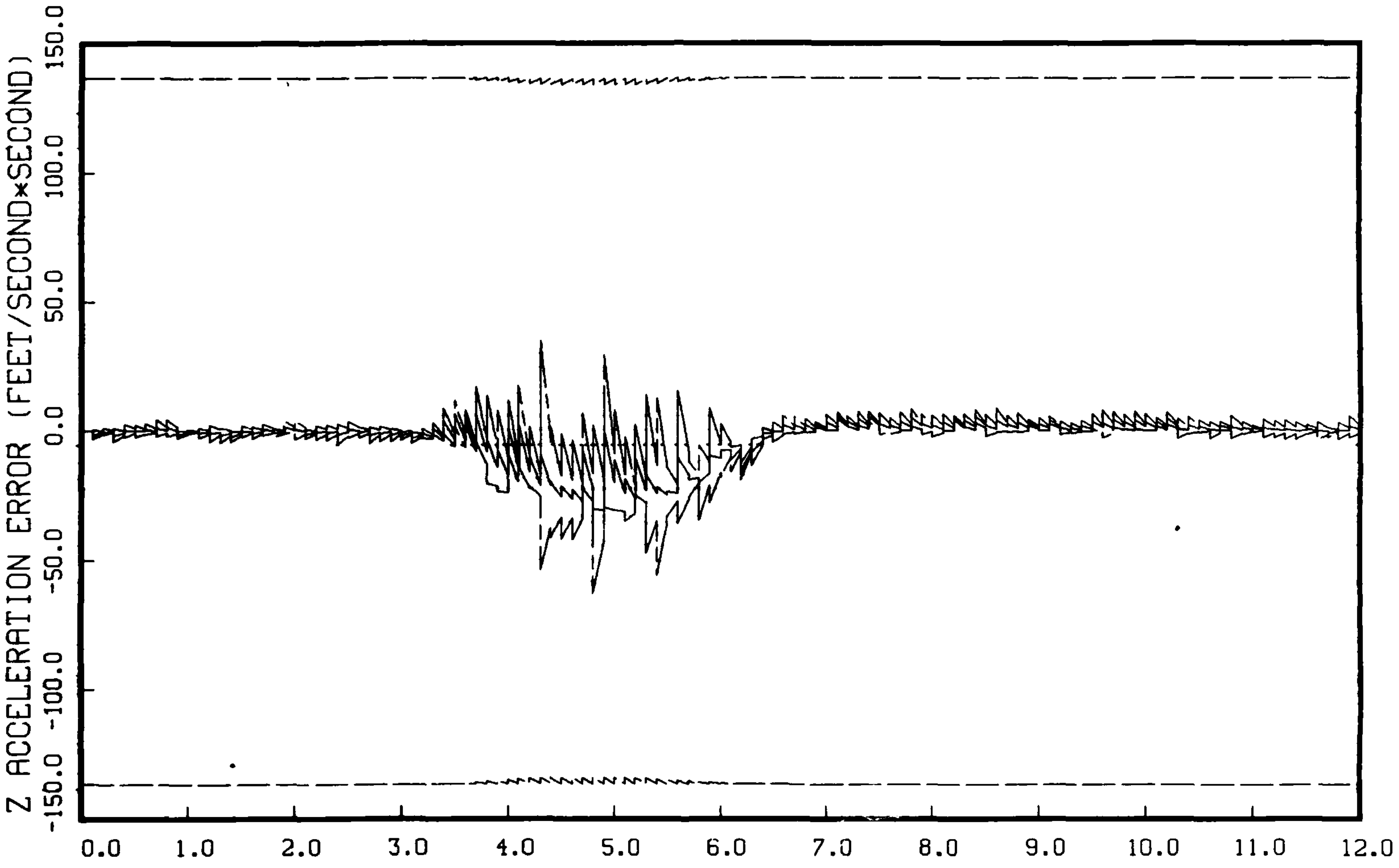
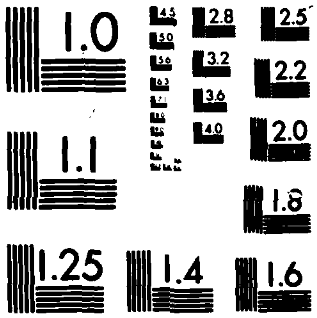


Figure G.2.5.i

STATE 9, Q(1)-Q(2)-Q(3)-373250., TAU(1)=.10,TAU(2-3)-.10, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

AD-A164 034   PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT   4/5
MANEUVERING TARGET..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI..   R B ANDERSON
UNCLASSIFIED   DEC 85 AFIT/GE/ENG/85D-2   F/G 19/5   NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

70 PS3 1 - 1 SOFEPL DATE AND TIME - 85/10/03. 11.49.02. SOFE DATE AND TIME - 85/10/03. 10.44.10. WPAFB, DISSPLA, SOFEPL VERSION 2.2

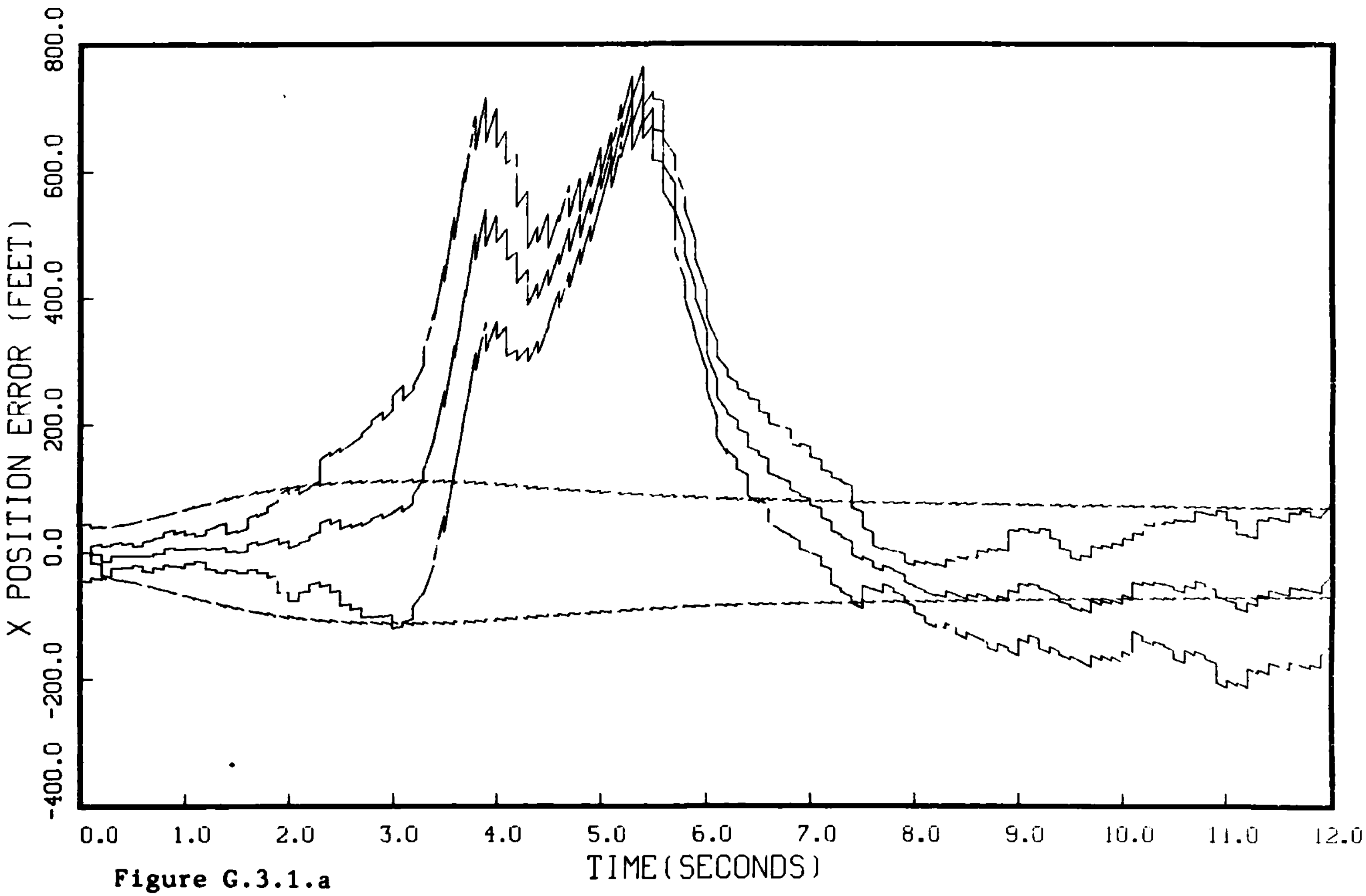


Figure G.3.1.a

STATE 1, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

2 SOFEPL DATE AND TIME - 85/10/03. 11.49.02. SOFE DATE AND TIME - 85/10/03. 10.44.10. WPAFB, DISSPLA, SOFEPL VERSION 2.2

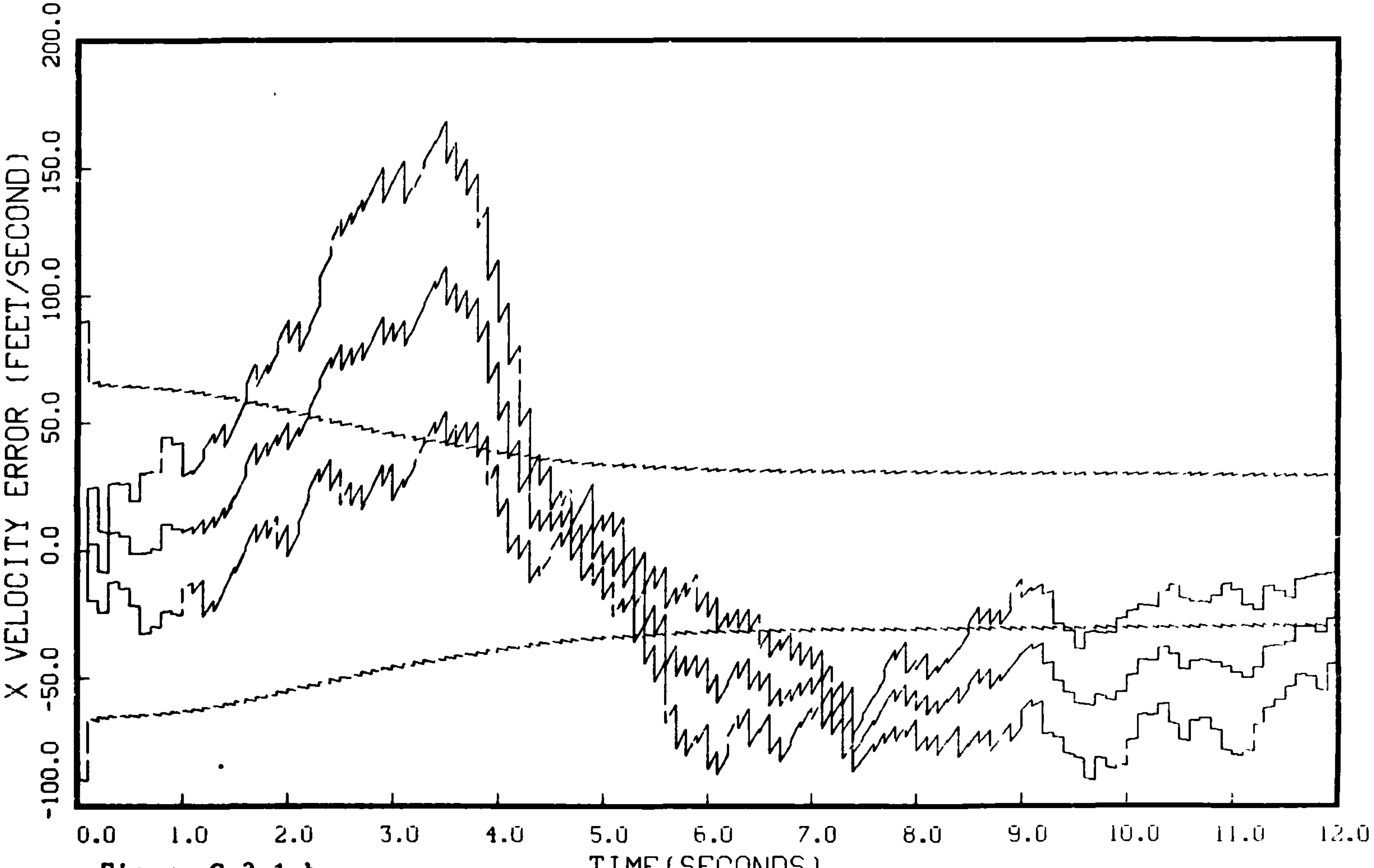


Figure G.3.1.b

STATE 2, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

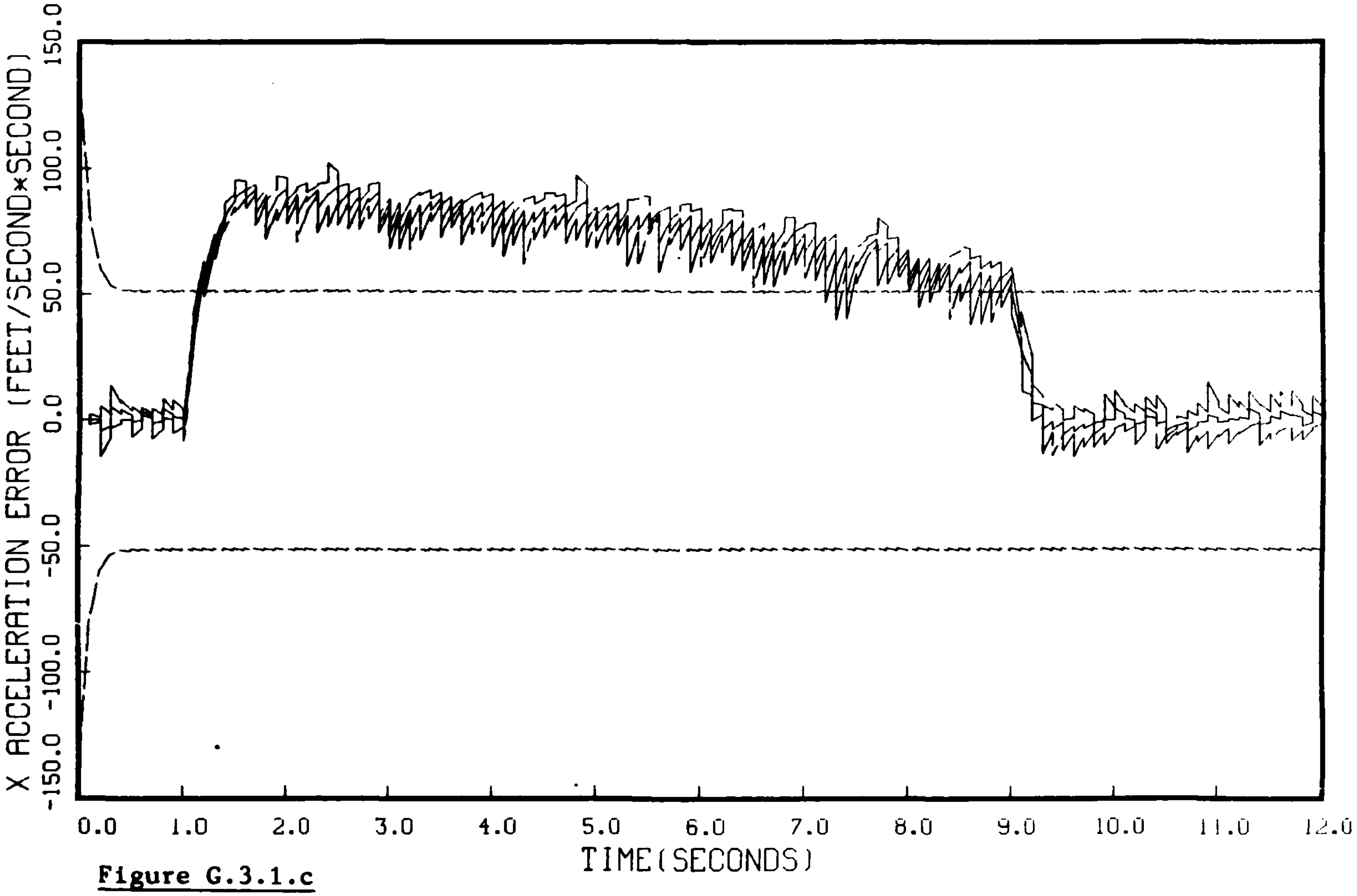


Figure G.3.1.c

STATE 3, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

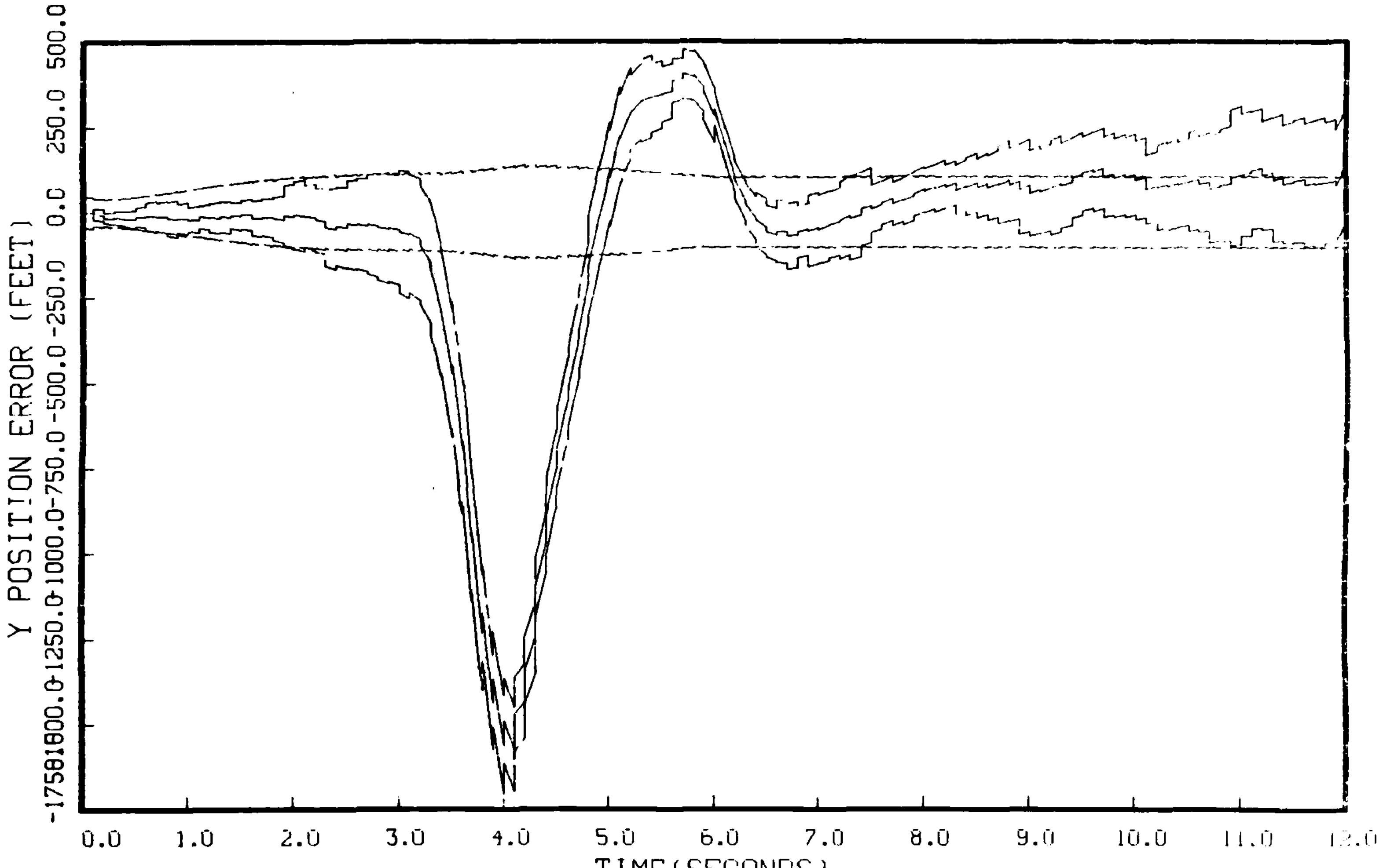


Figure G.3.1.d

STATE 4, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

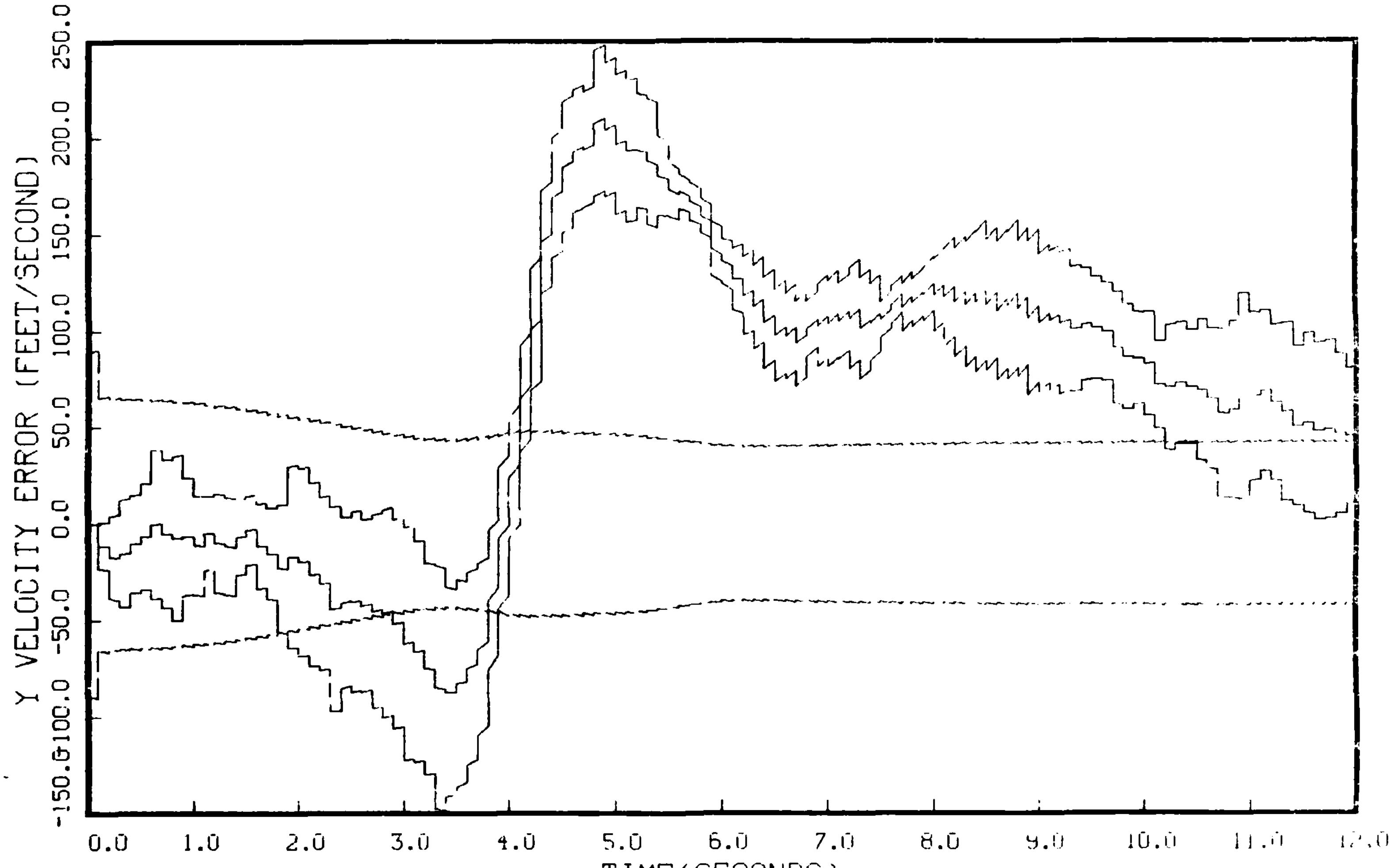


Figure G.3.1.e

STATE 5, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

6 SOFEPL DATE AND TIME - 85/10/03. 11.49.02. SOFE DATE AND TIME - 85/10/03. 10.44.10. WPAFB, DISSPLA, SOFEPL VERSION 2.2

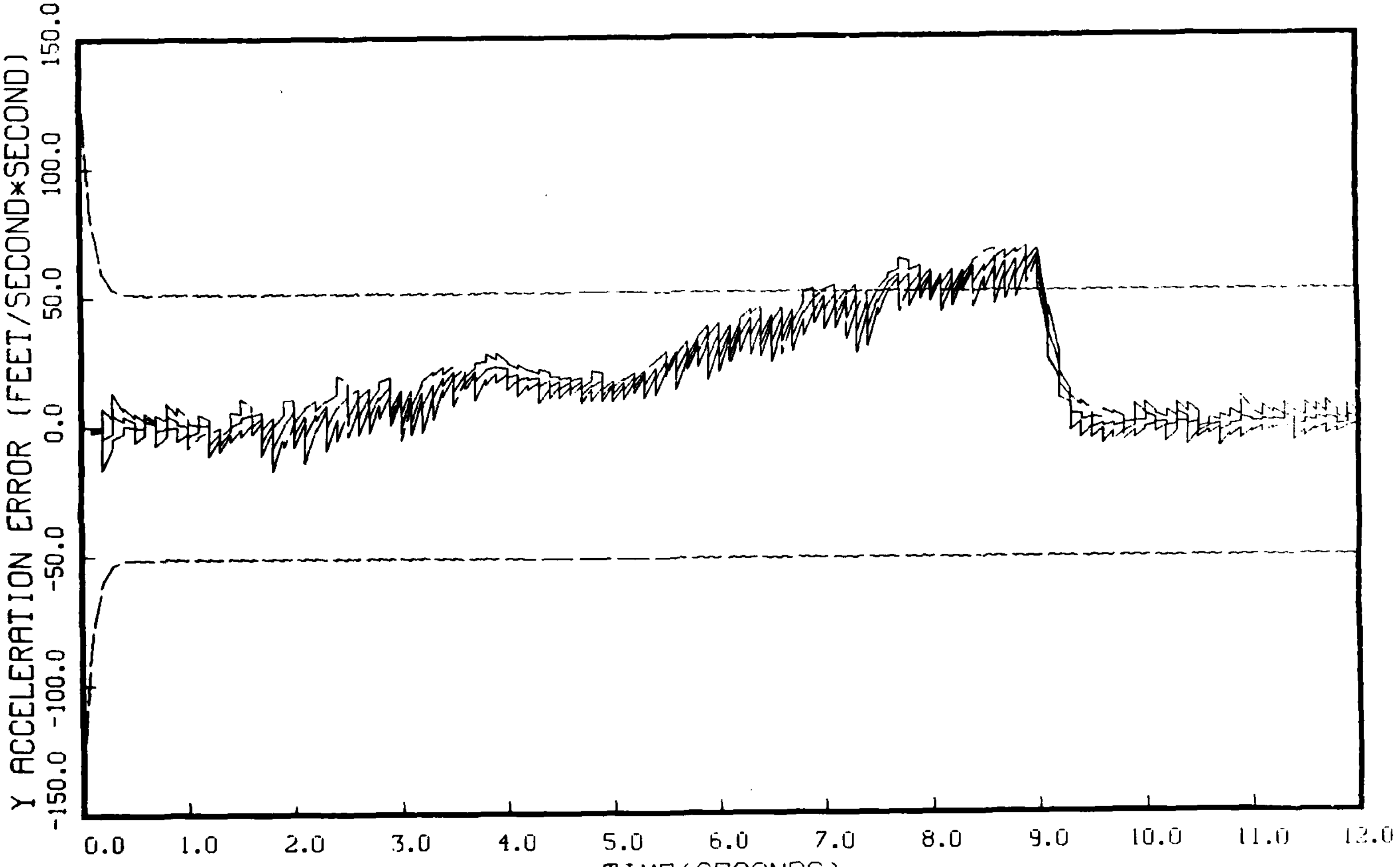


Figure G.3.1.f

STATE 6, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE-.1, 5 RUNS

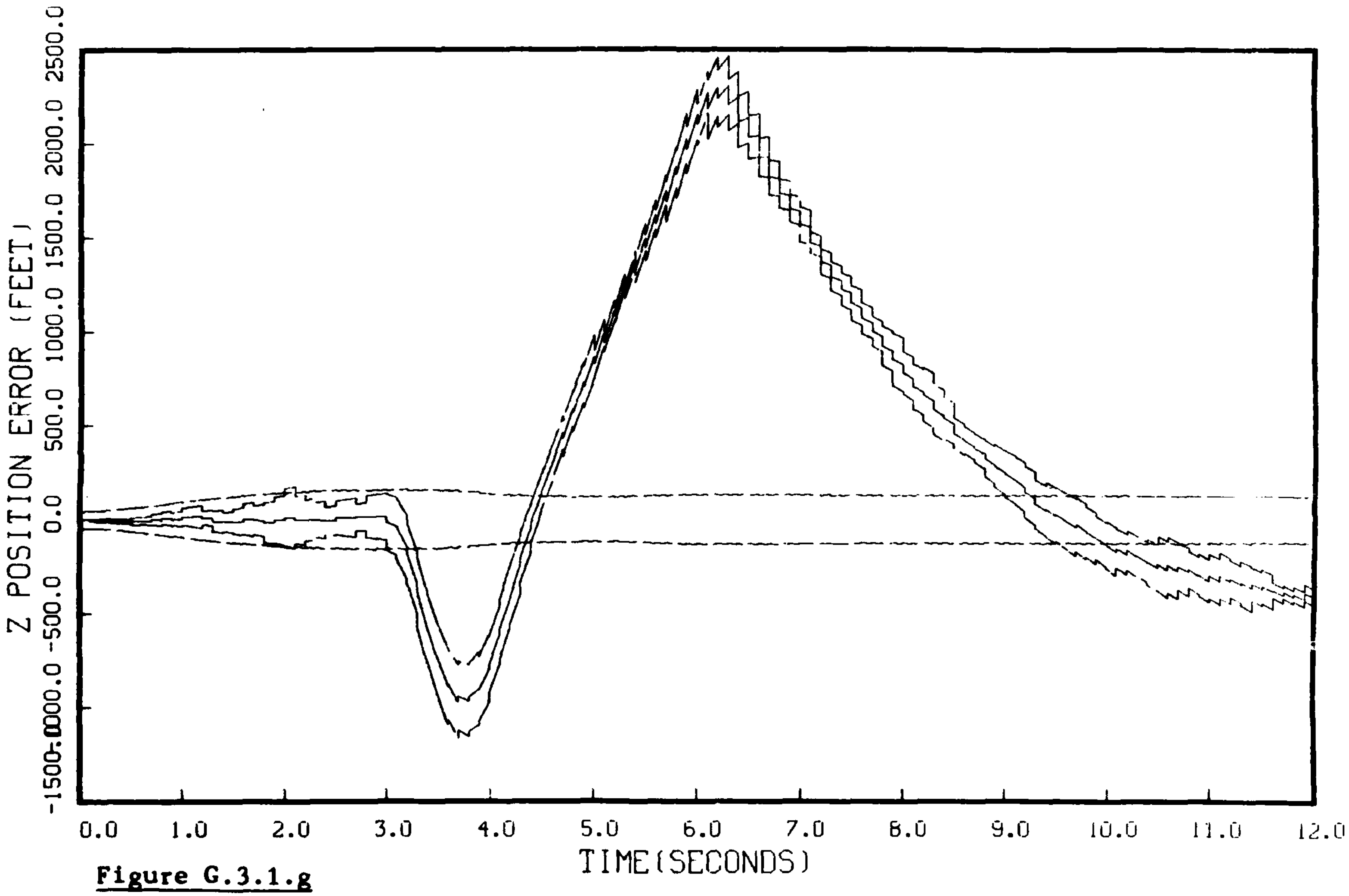


Figure G.3.1.g

STATE 7, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

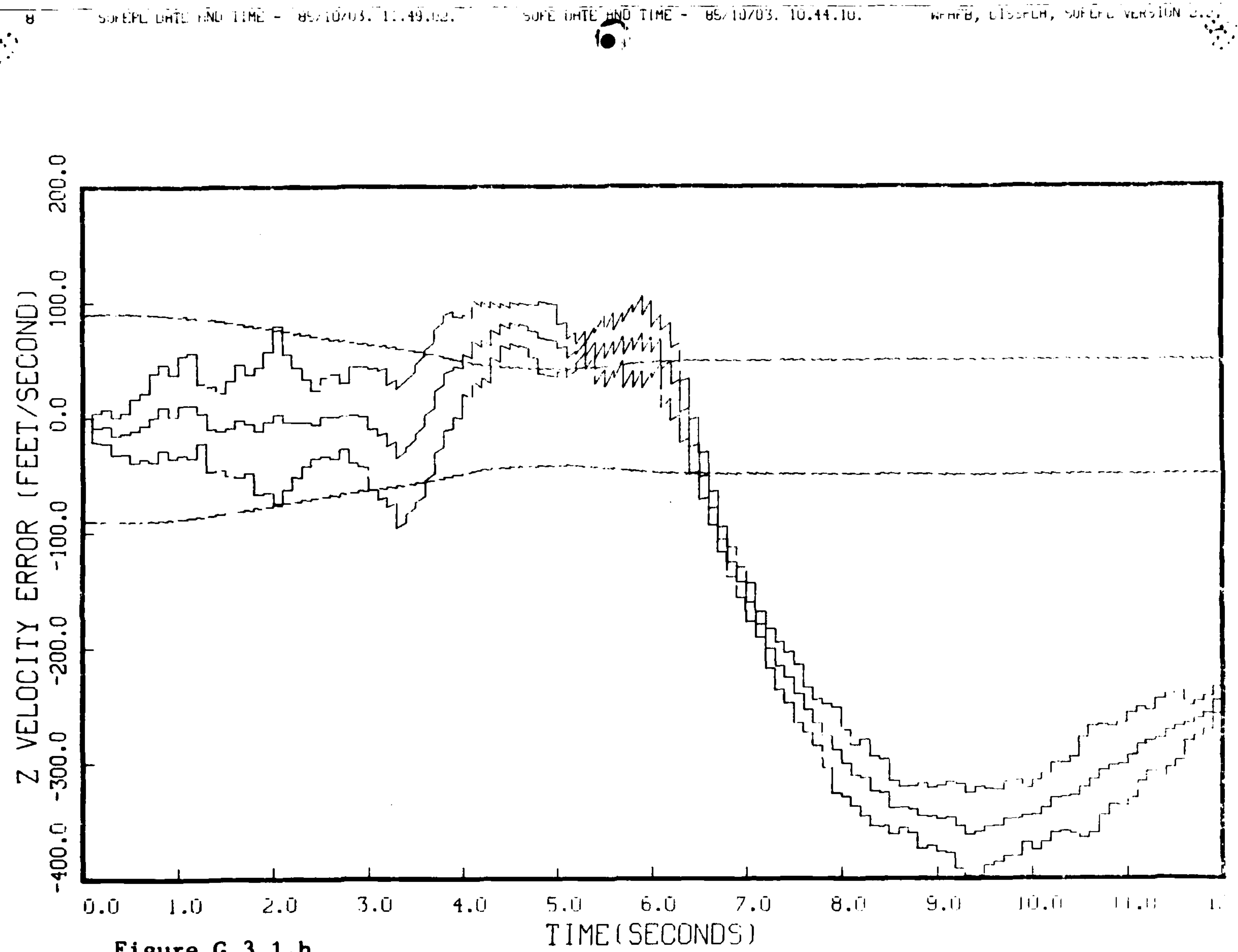


Figure G.3.1.h

STATE 8, Q(1)=Q(2)=Q(3)=37325., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

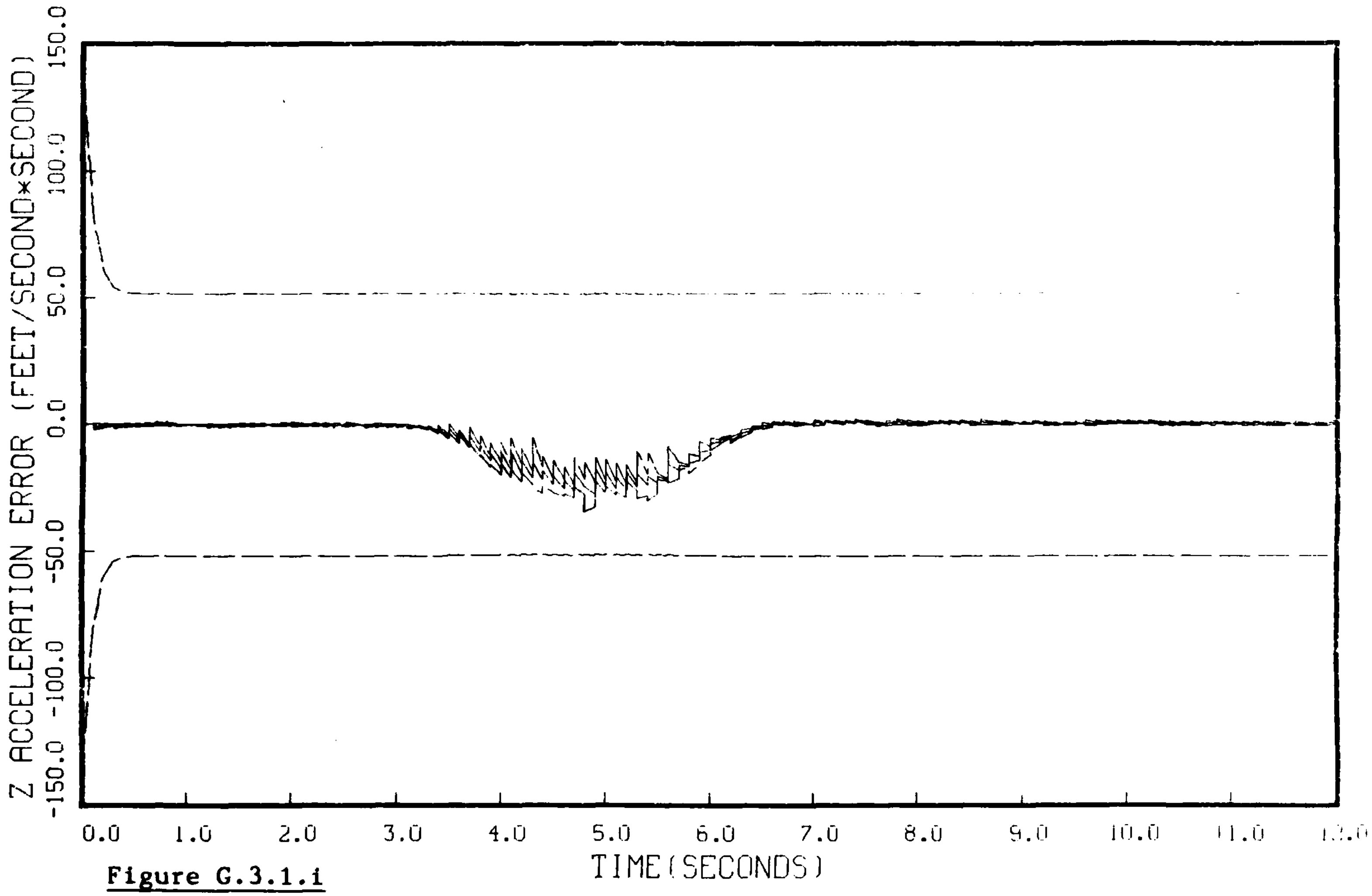


Figure G.3.1.i

STATE 9, Q(1)-Q(2)-Q(3)-37325., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

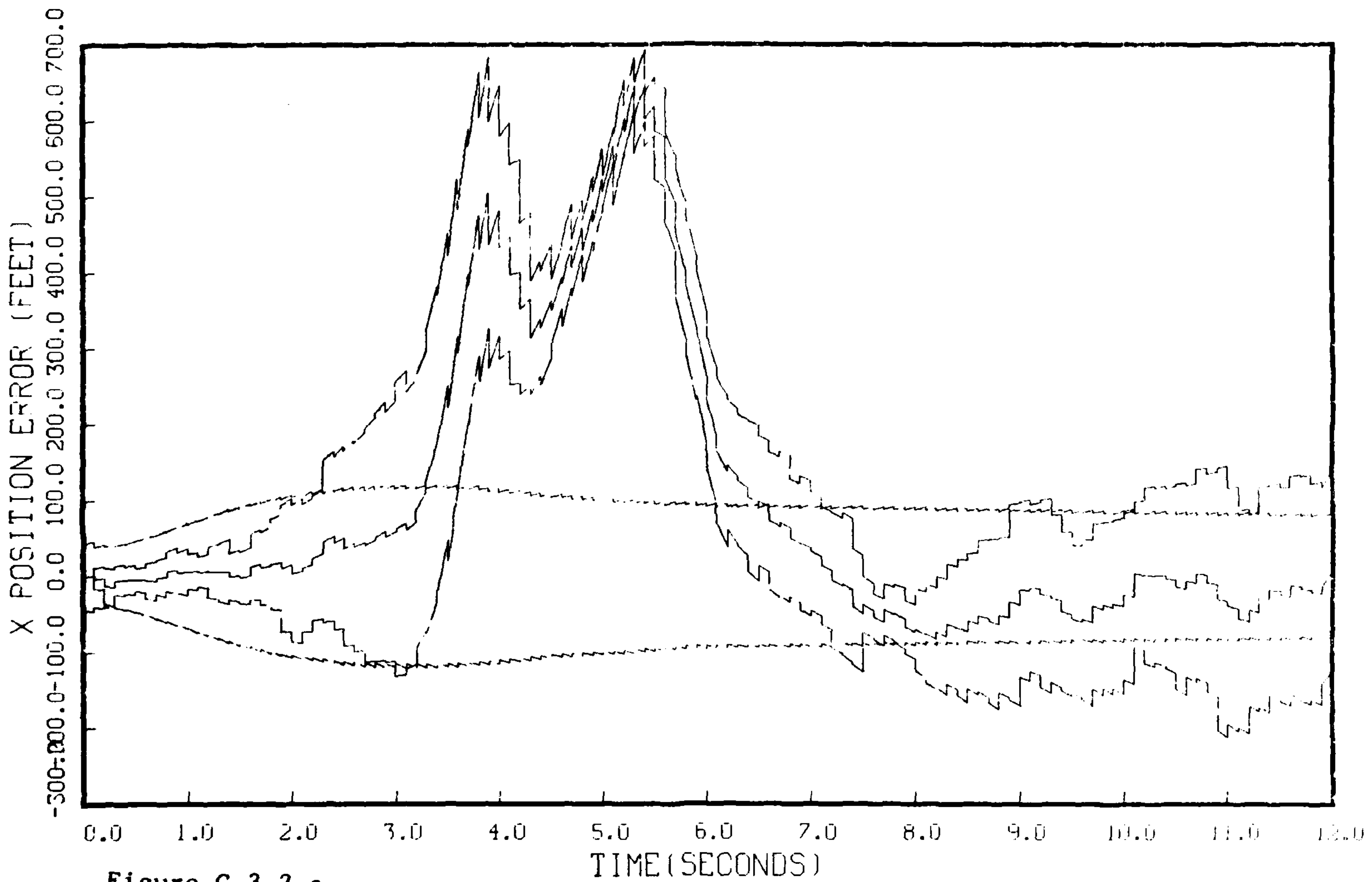


Figure G.3.2.a

STATE 1, Q(1)-Q(2)-Q(3)-149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

2 SUPEPL DATE AND TIME - 85/10/03. 14.06.42. SUPE DATE AND TIME - 85/10/03. 12.17.38. WPAFB, DISSPLA, SUPEPL VERSION 2.

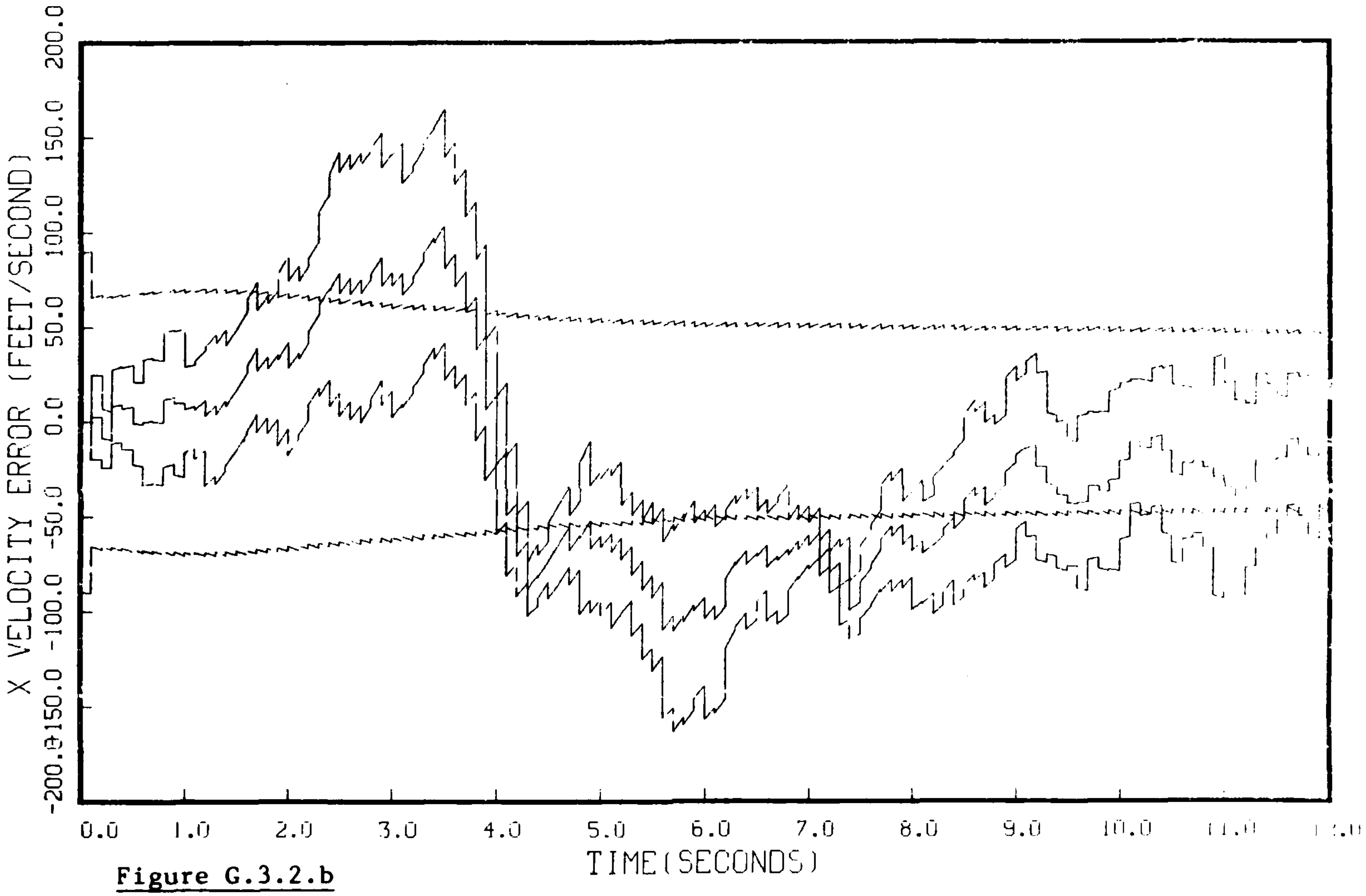


Figure G.3.2.b

STATE 2, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE=.1, 5 RUNS

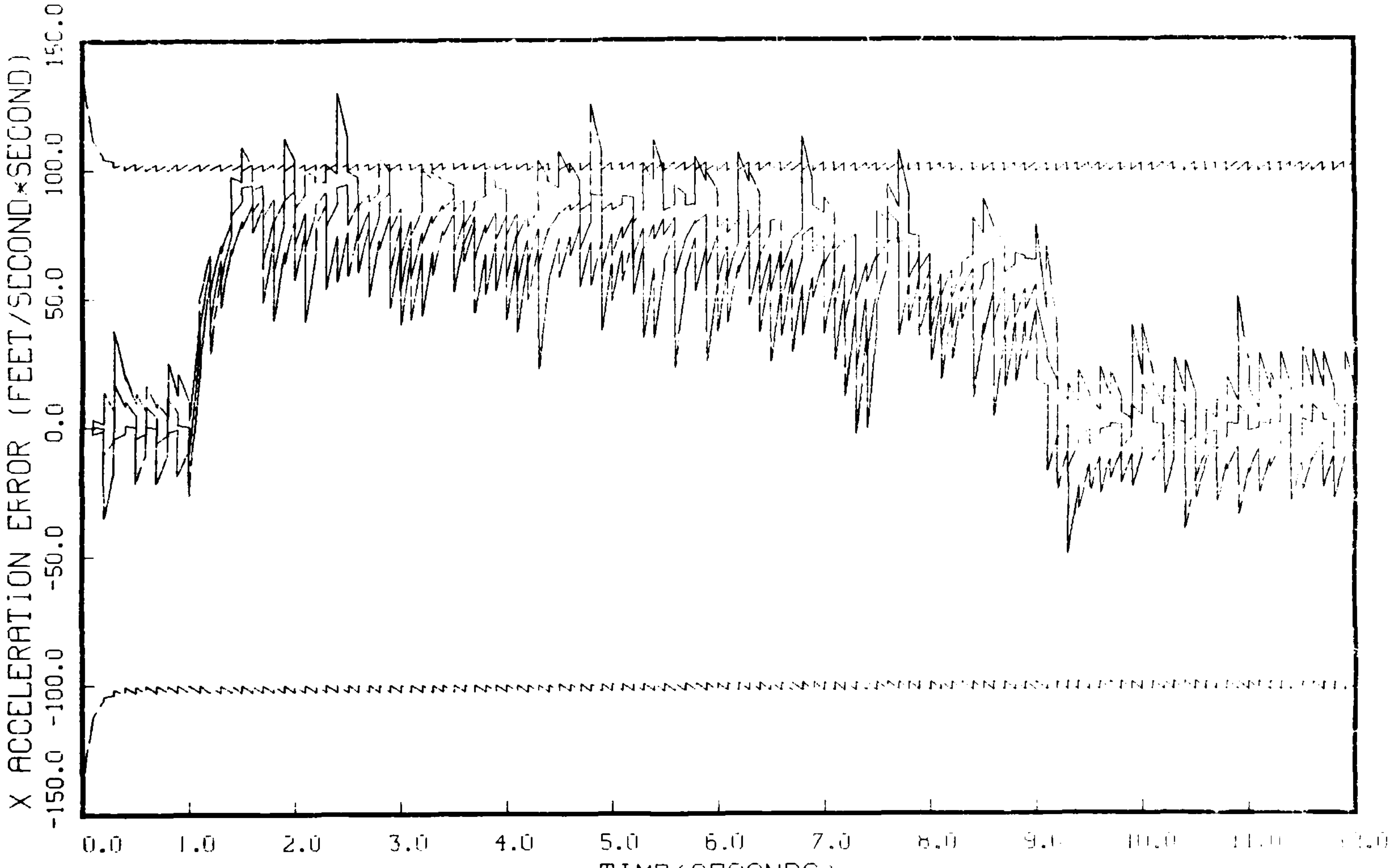


Figure G.3.2.c

STATE 3, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUN

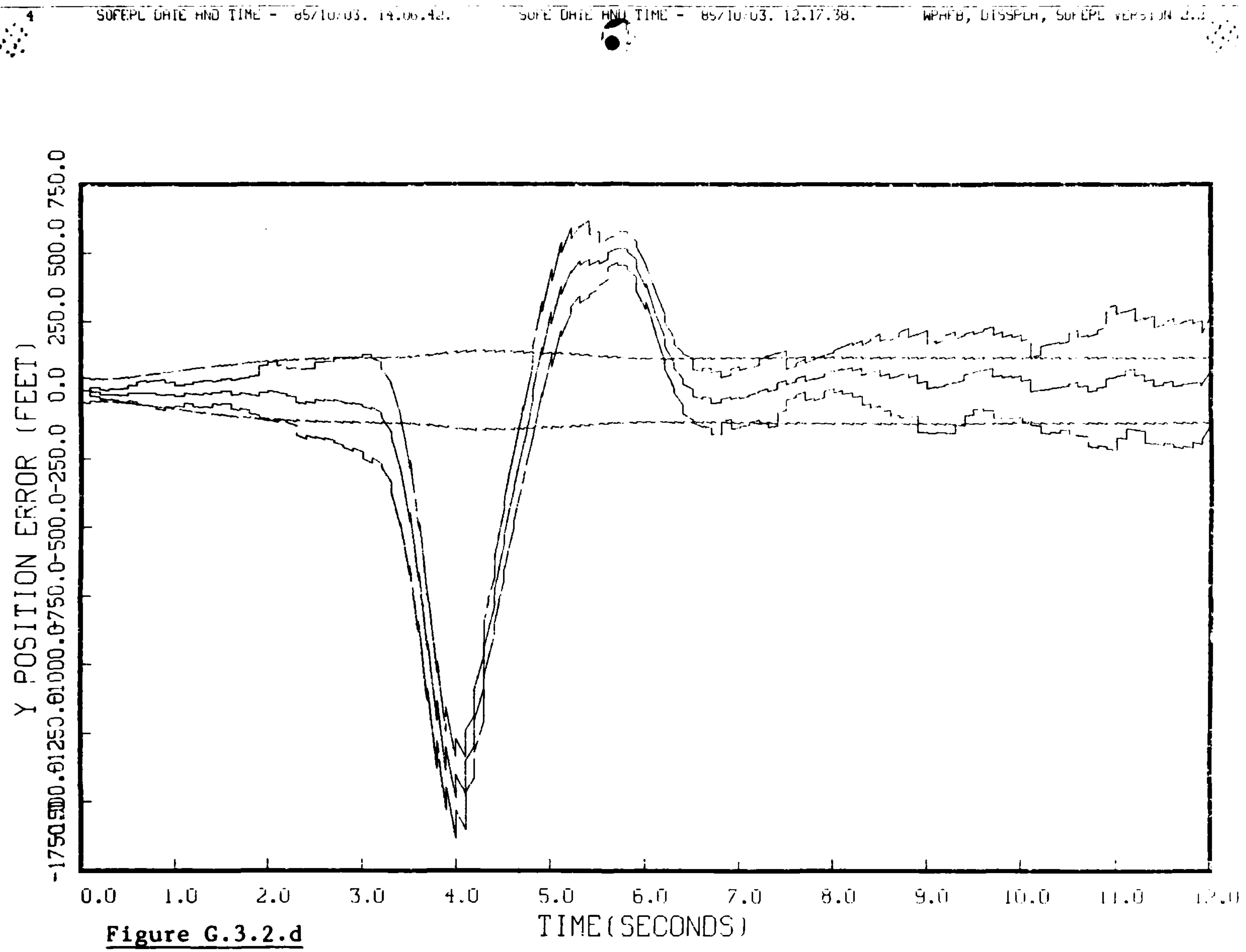


Figure G.3.2.d

STATE 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

5 SOFEPL DATE AND TIME - 85/10/03. 14.06.42. SOFE DATE AND TIME - 85/10/03. 12.17.38. WPAFB, DISSPLA, SOFEPL VERSION 2.2

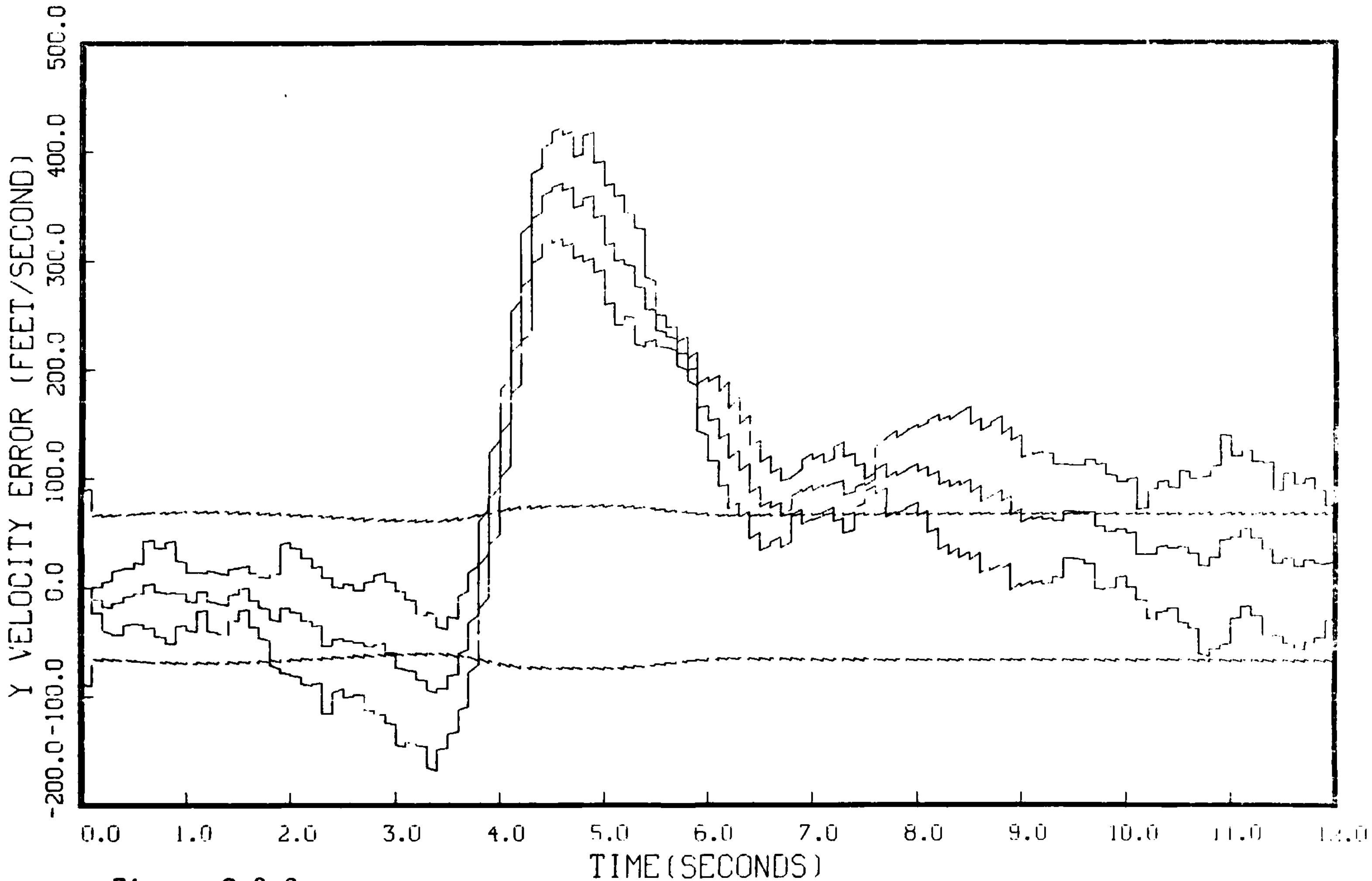


Figure G.3.2.e

STATE 5, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
AFO-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

6 SOFEPL DATE AND TIME - 85/10/03. 14.06.42. SOFE DATE AND TIME - 85/10/03. 12.17.38. WPAFB, DISSPLA, SOFEPL VERSION 2.2

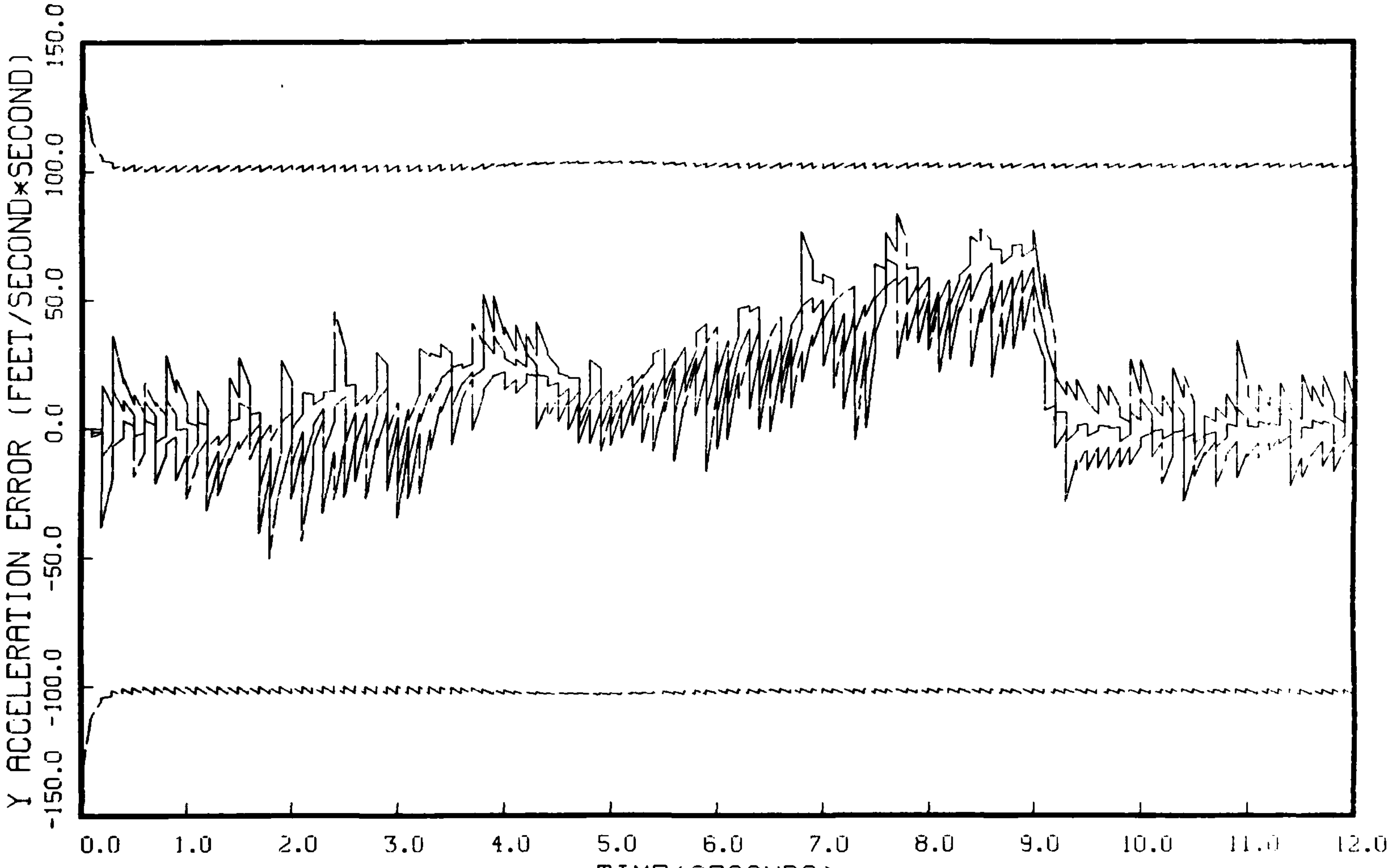


Figure G.3.2.f

STATE 6, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
AFO-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

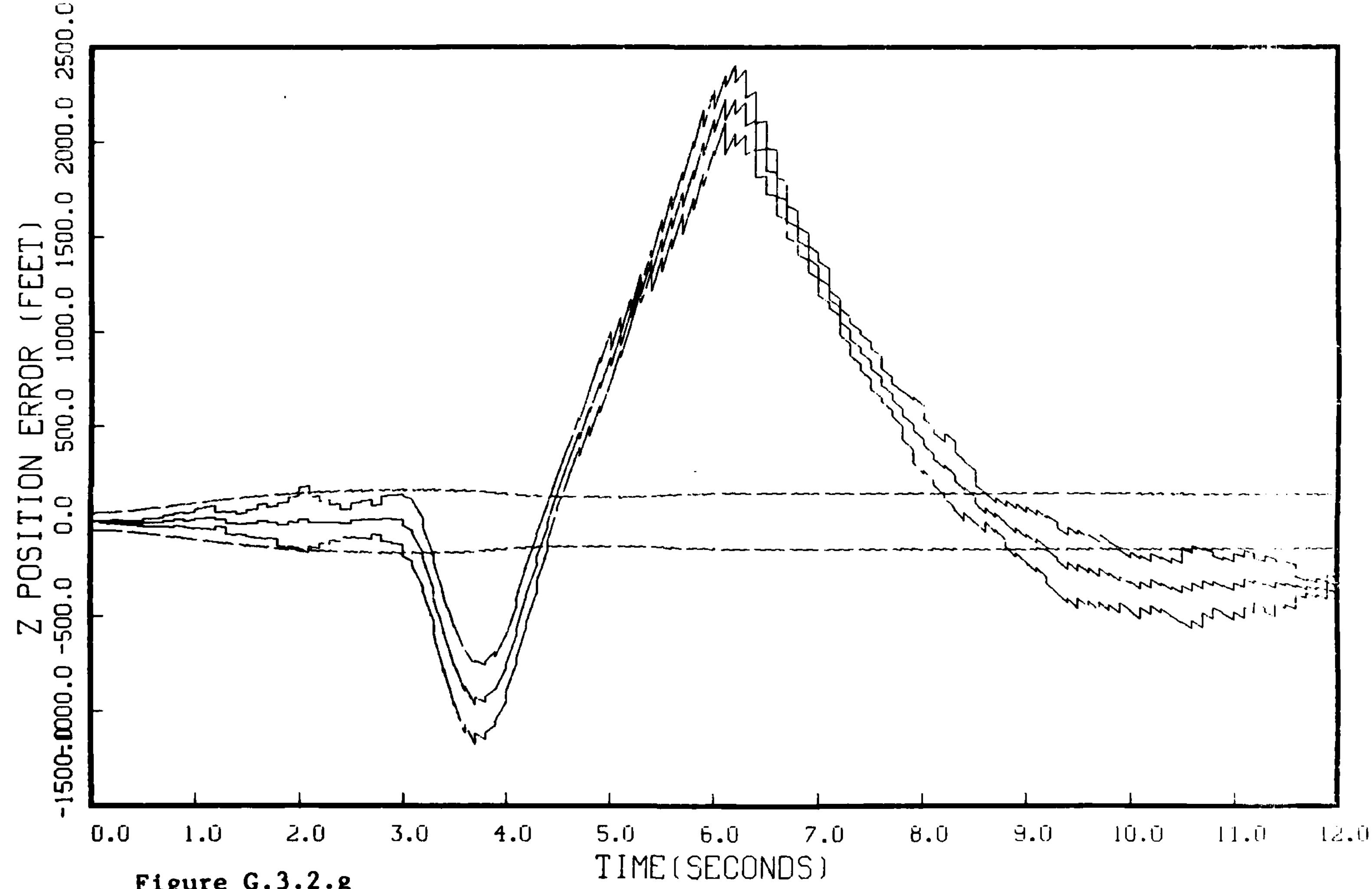


Figure G.3.2.g

STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

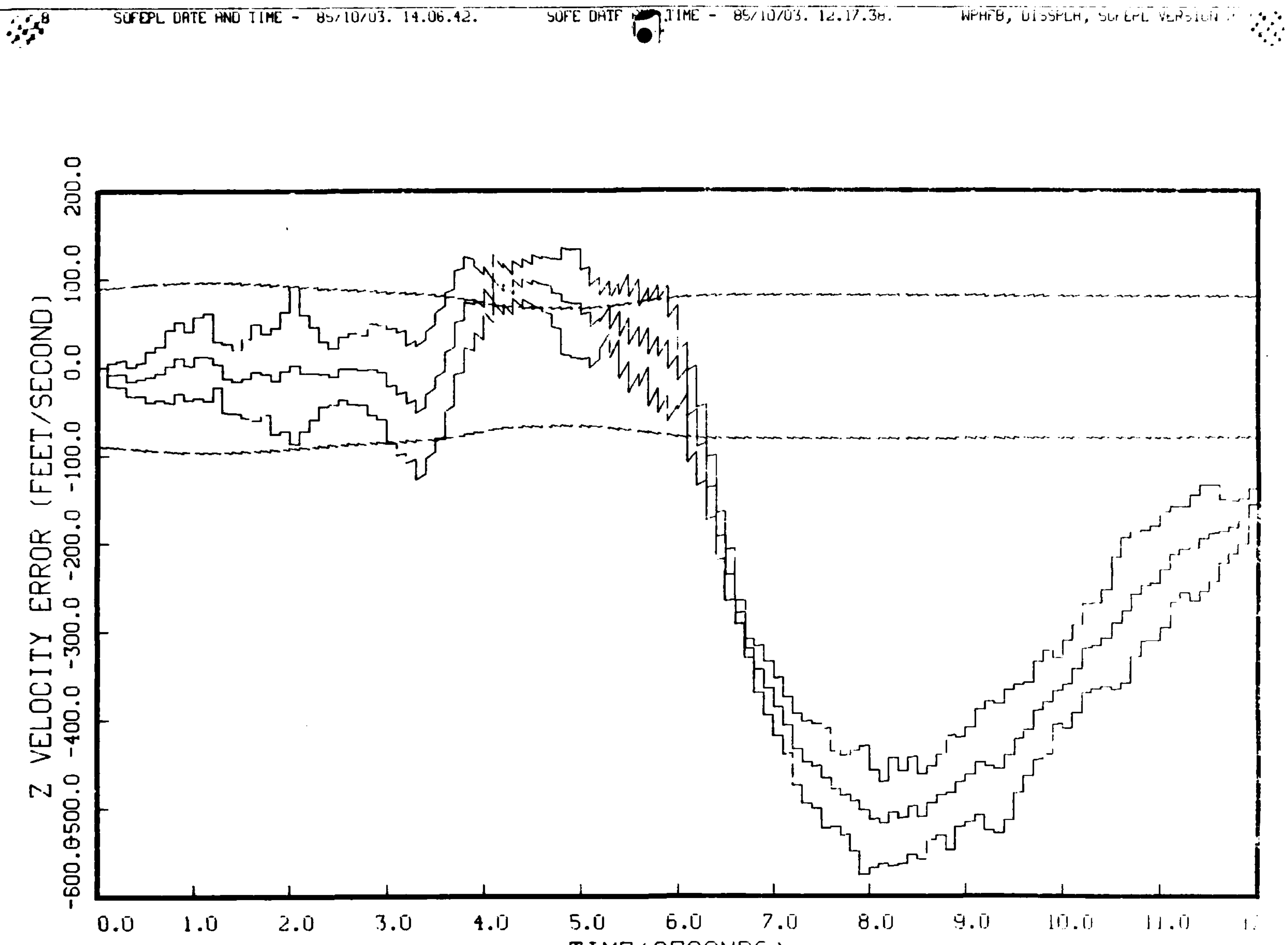


Figure G.3.2.h

STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

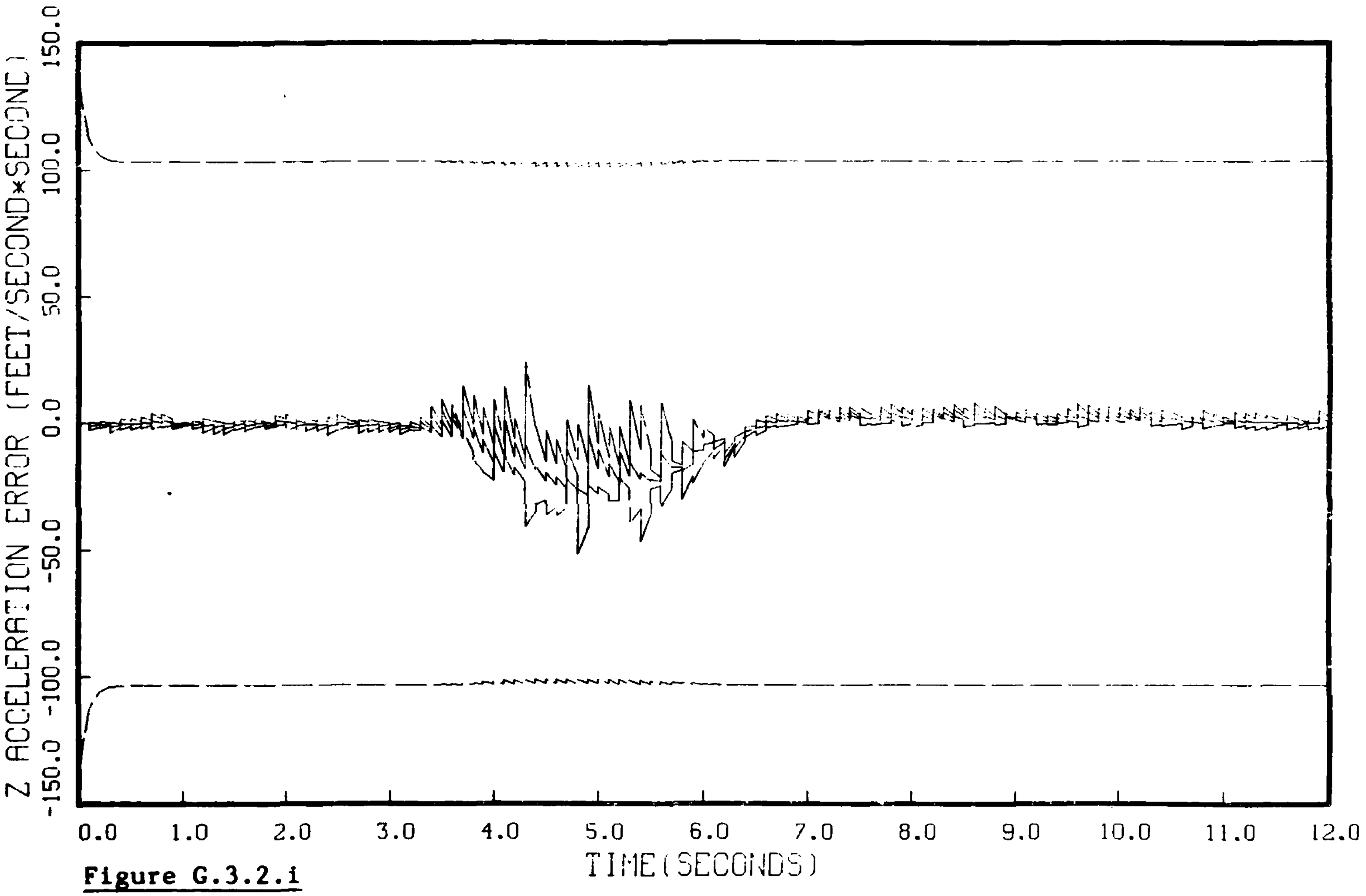


Figure G.3.2.i

STATE 9, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

10 SOFEPL DATE AND TIME - 85/10/03. 14.06.42. SOFE DATE AND TIME - 85/10/03. 12.17.38. WPAFB, DISSPLA, SOFEPL VERSION 2.2

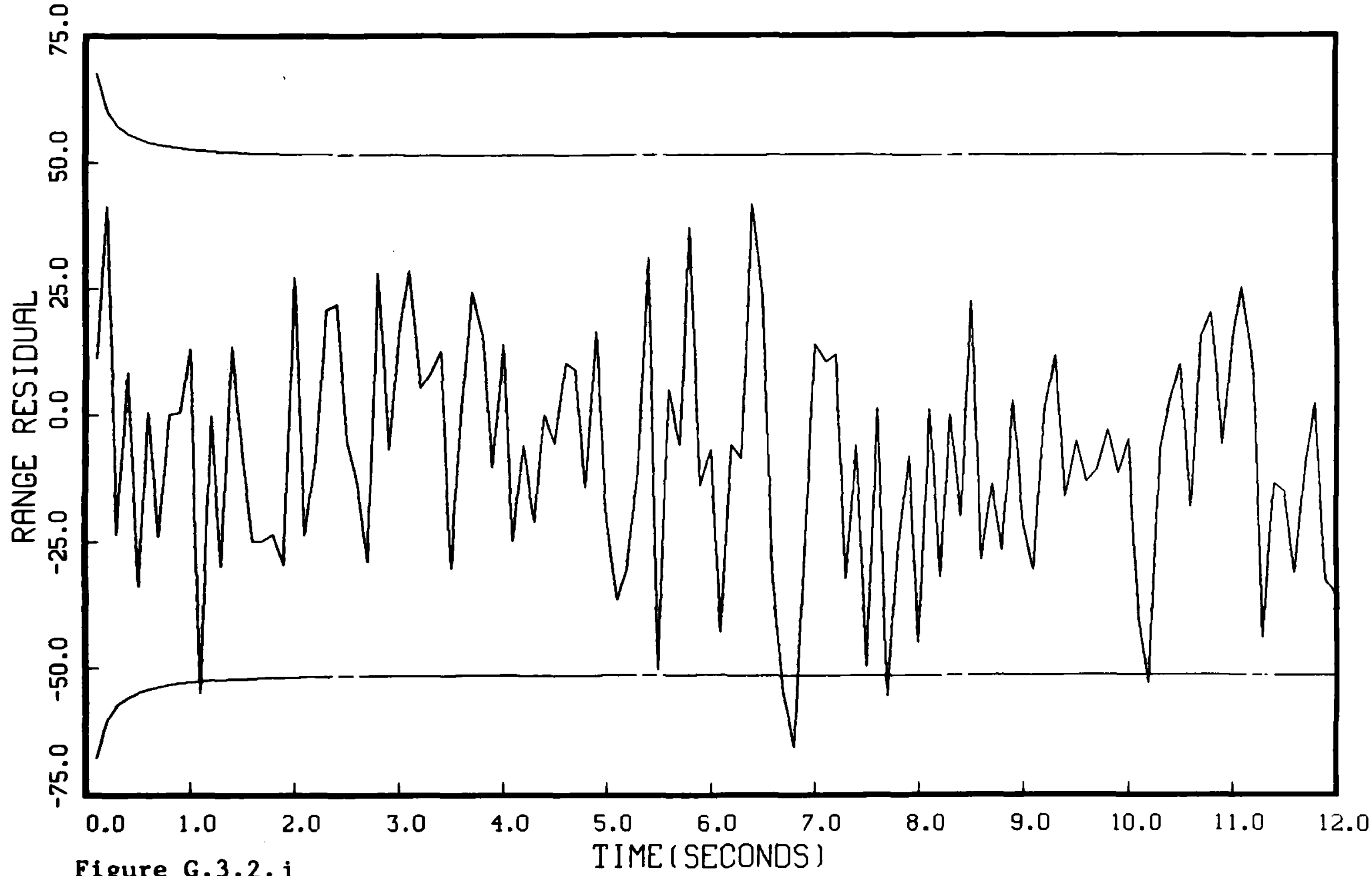


Figure G.3.2.j

MEAS 1, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

11 SOFEPL DATE AND TIME - 85/10/03. 14.06.42. SOFE DATE AND TIME - 85/10/03. 12.17.38. WPAFB, DISSPLA, SOFEPL VERSION 2.2

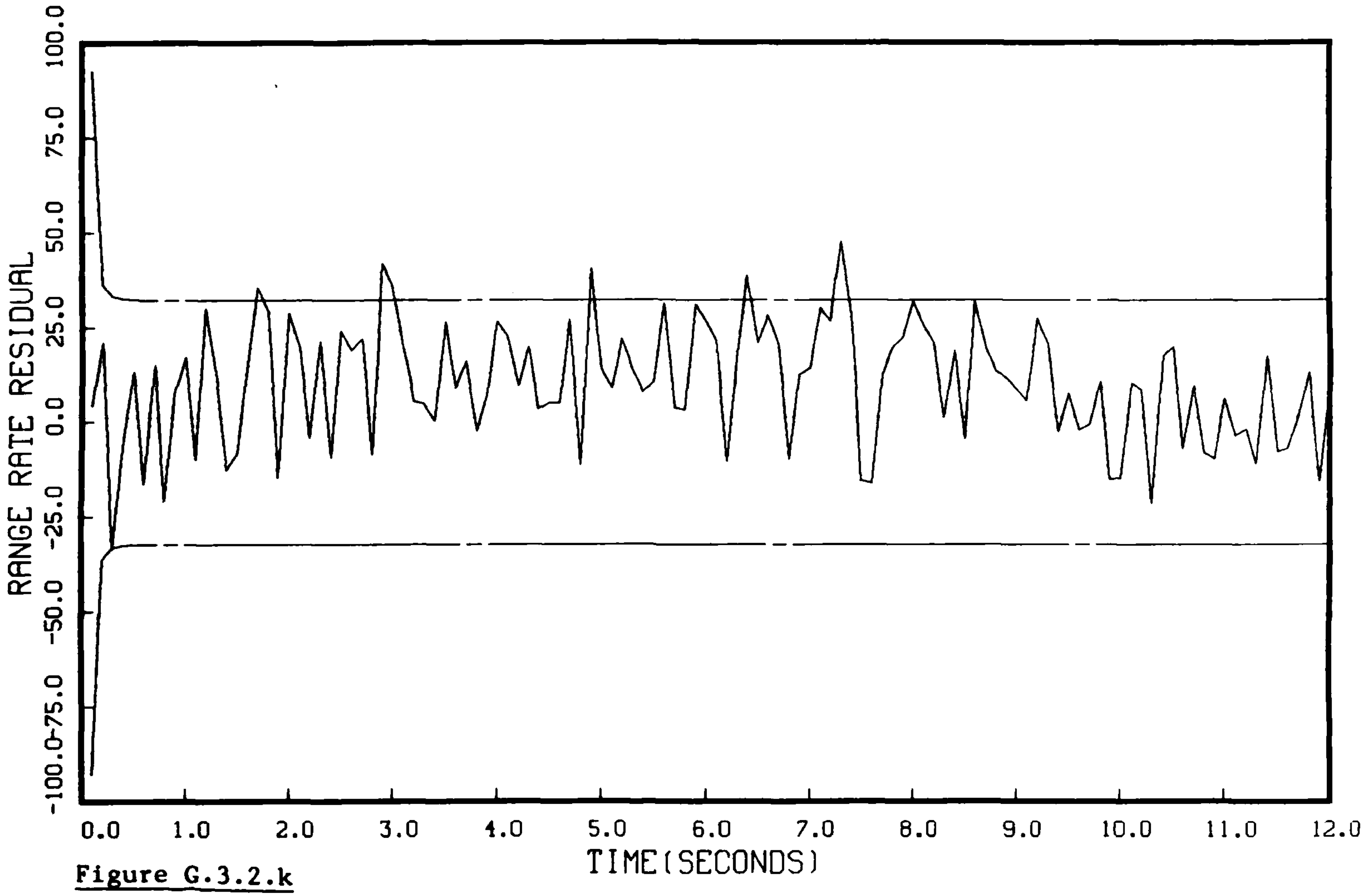


Figure G.3.2.k

MEAS 2, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

12 SOFEPL DATE AND TIME - 85/10/03. 14.06.42. SOFE DATE AND TIME - 85/10/03. 12.17.38. WPAFB, DISSPLA, SOFEPL VERSION 2.2

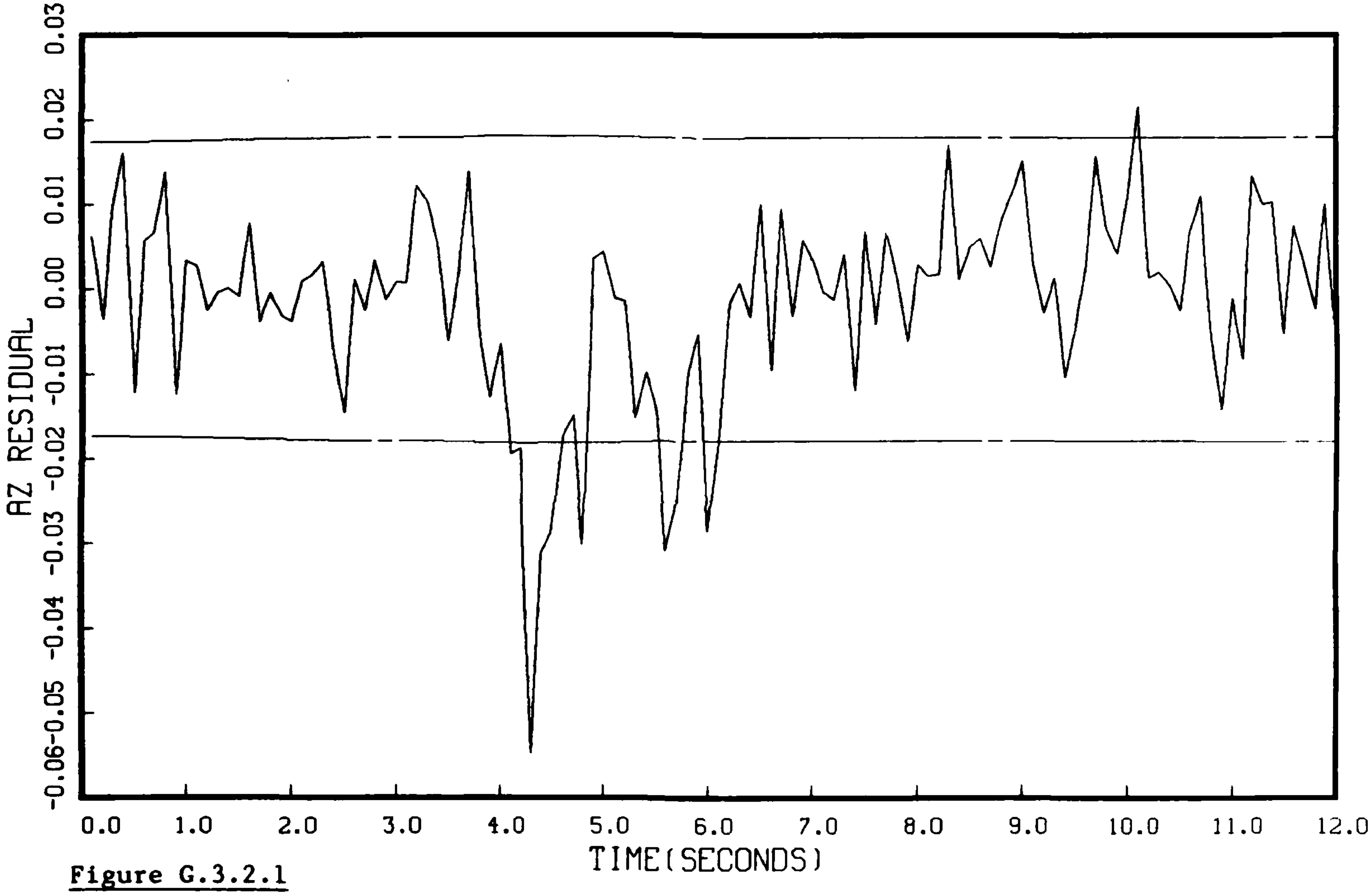


Figure G.3.2.1

MEAS 3, O(1)-O(2)-O(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

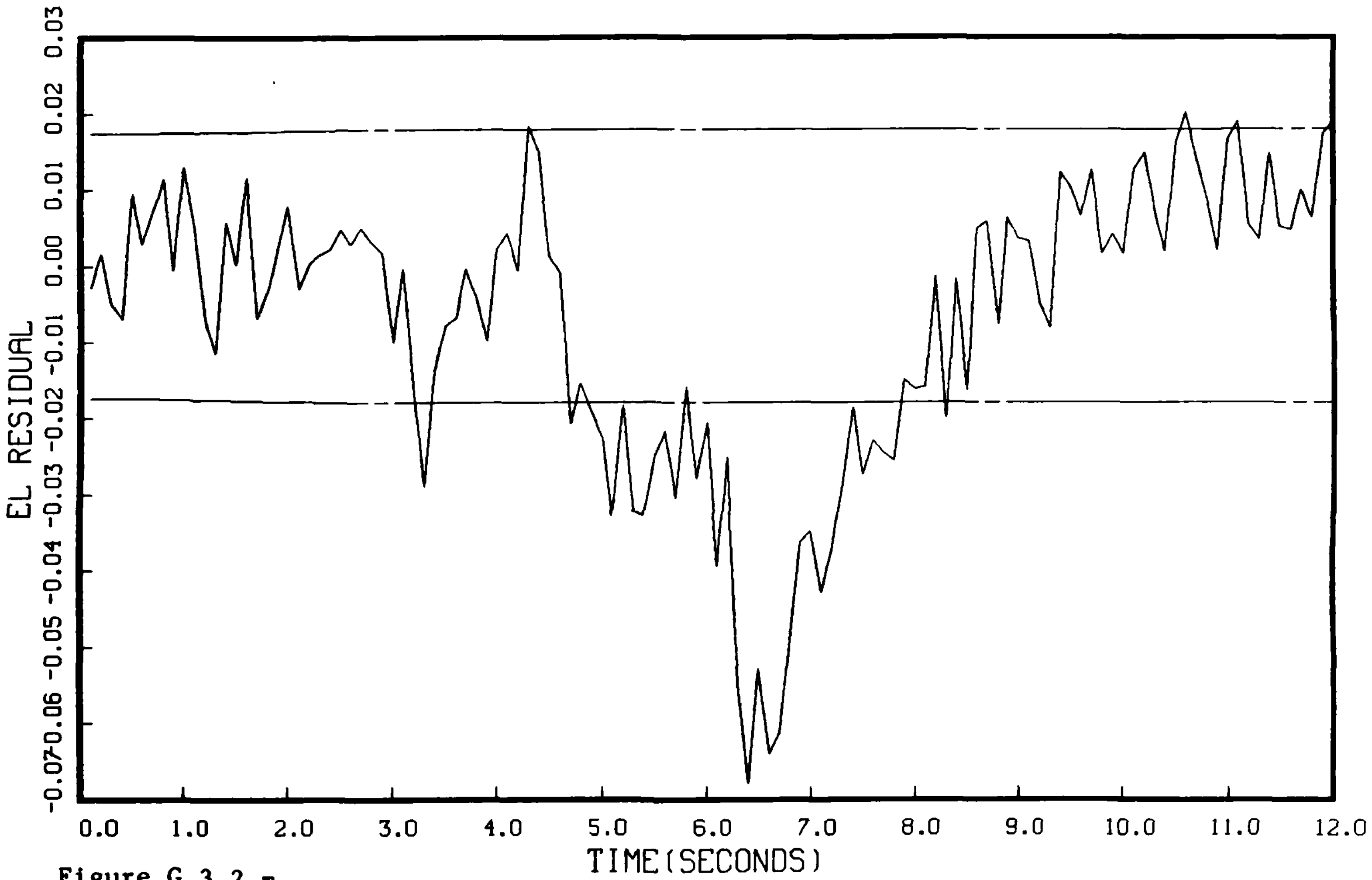


Figure G.3.2.m

MEAS 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

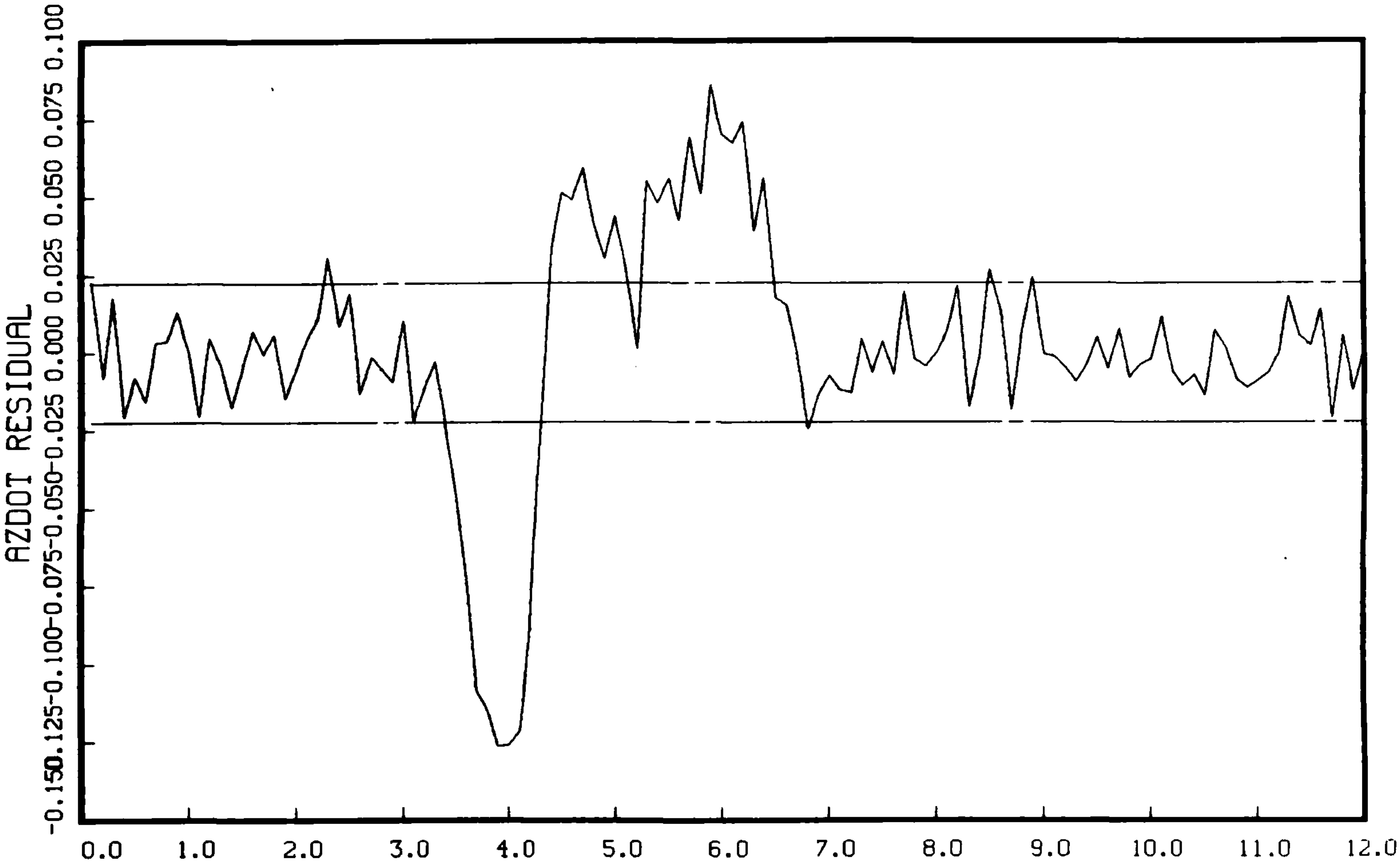


Figure G.3.2.n

MEAS 5, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

15 SOFEPL DATE AND TIME - 85/10/03. 14.06.42. SOFE DATE AND TIME - 85/10/03. 12.17.38. WPAFB, DISSPLA, SOFEPL VERSION 2.2

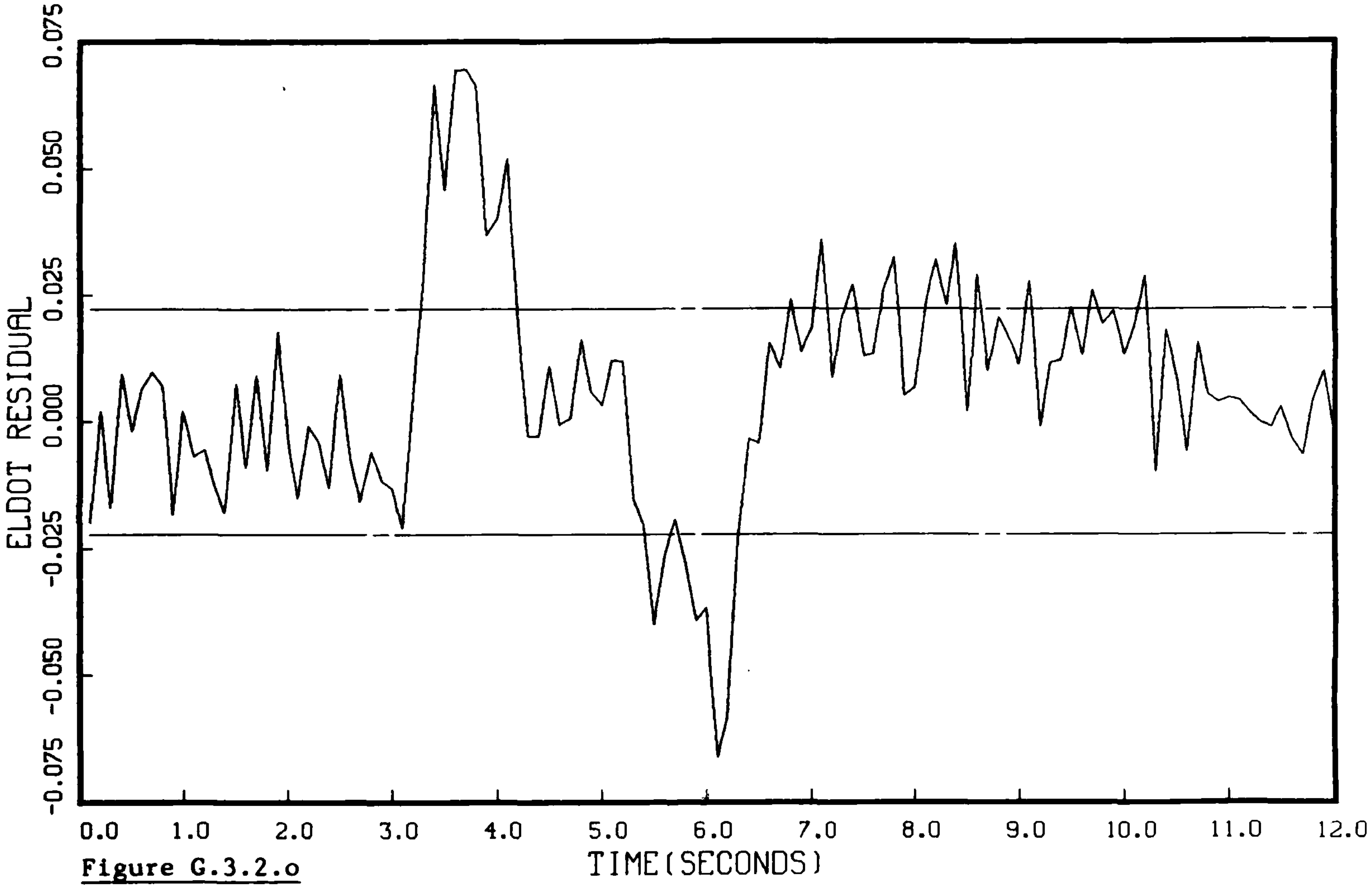


Figure G.3.2.o

MEAS 6, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

71 PS 3 1 SUFEPL DATE AND TIME - 85/10/03. 11.44.07. SUFE DATE AND TIME - 85/10/03. 10.58.02. NPHFB, DISSPLA, SUFEPL VERSION 2.?

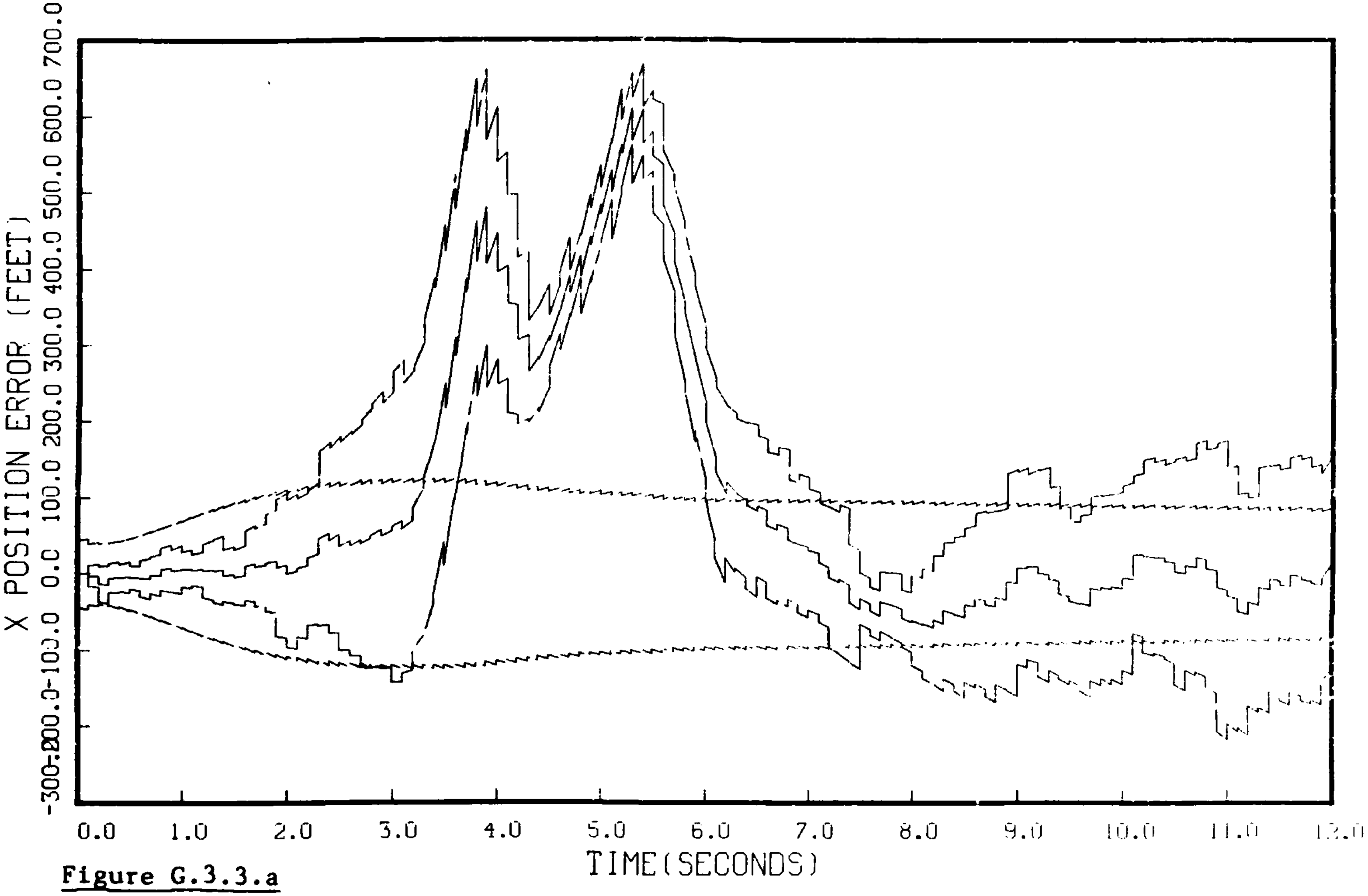


Figure G.3.3.a

STATE 1, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
AFO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

2 SOFEPL DATE AND TIME - 85/10/03. 11.44.07. SOFE DATE AND TIME - 85/10/03. 10.58.02. WPAFB, DISPLA, SOFEPL VERSION 2.2

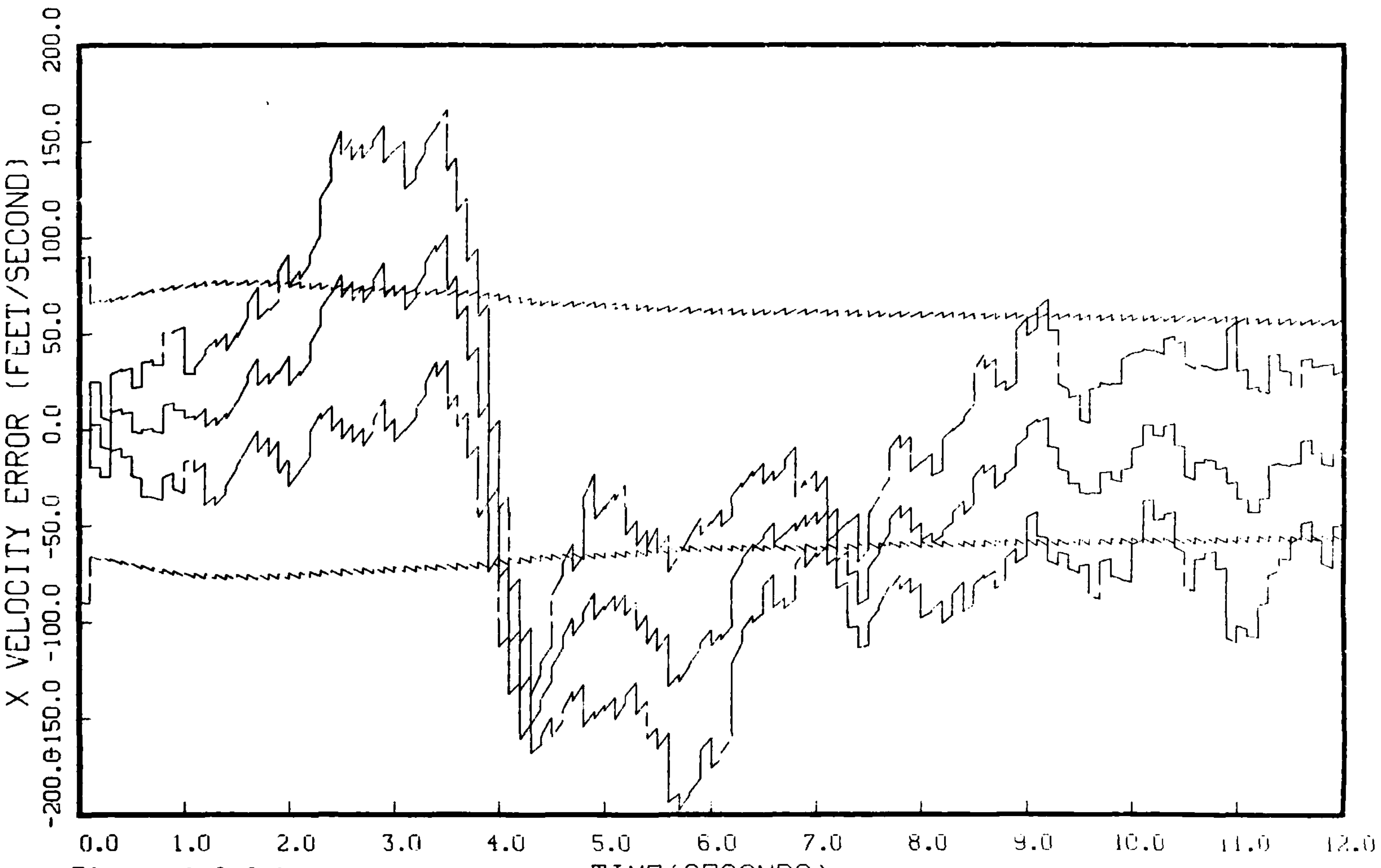


Figure G.3.3.b

STATE 2, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

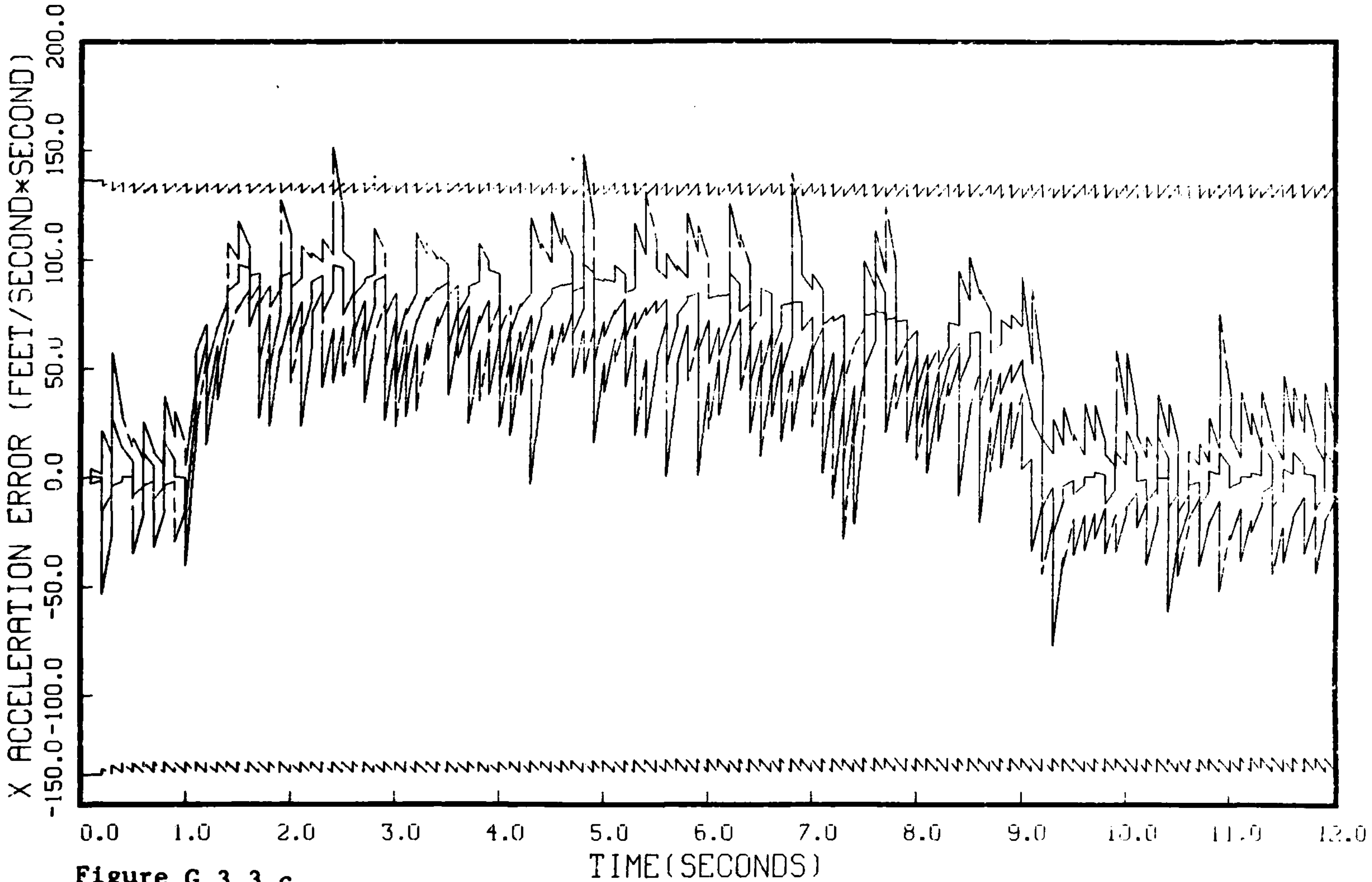


Figure G.3.3.c

STATE 3, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

4 SOFEPL DATE AND TIME - 85/10/03. 11.44.07. SOFE DATE AND TIME - 85/10/03. 10.58.02. WPAFB, DISSPLA, SOFEPL VERSION 2.2

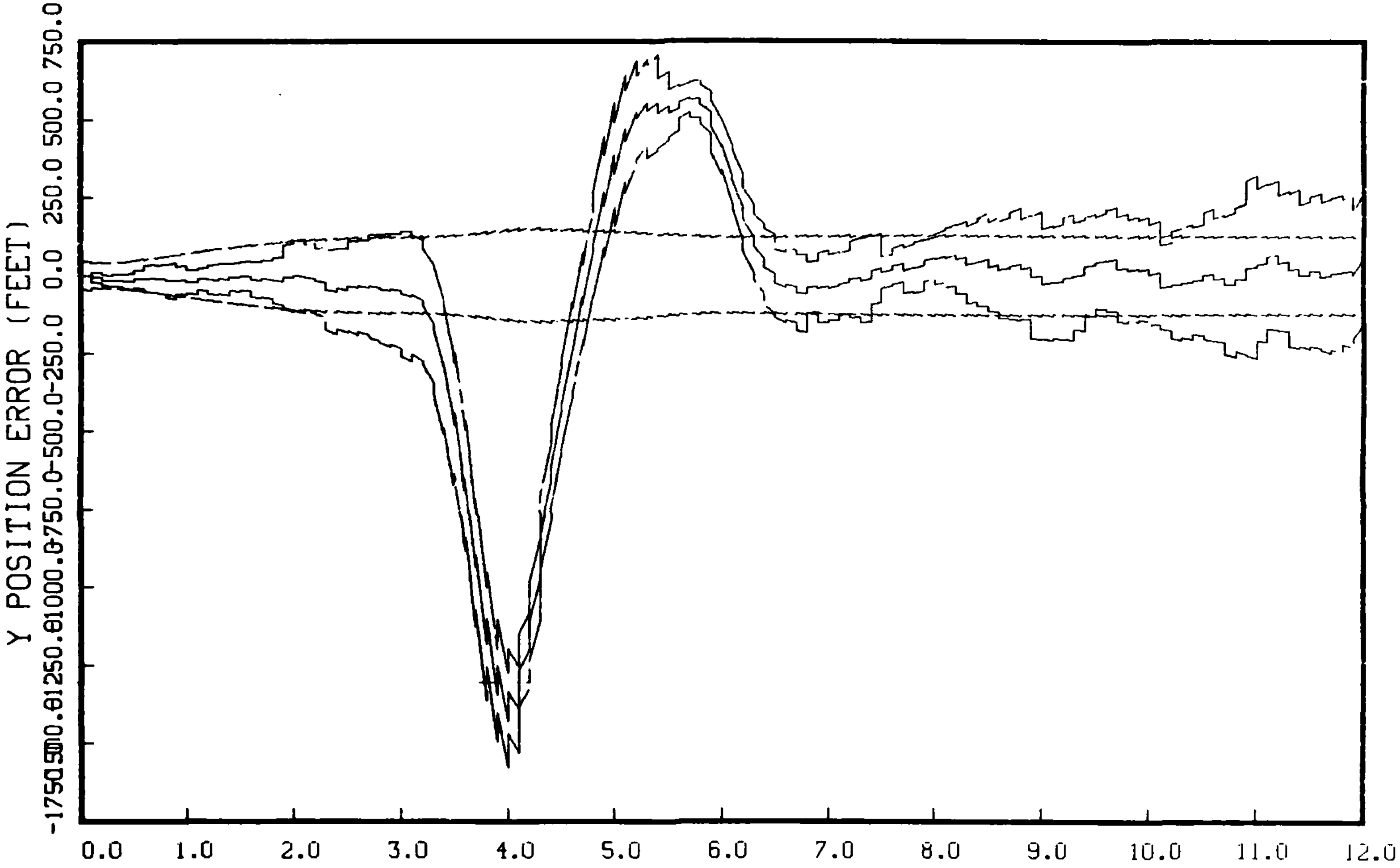


Figure G.3.3.d

STATE 4, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

5 SOFEPL DATE AND TIME - 85/10/03. 11.44.07. SOFE DATE AND TIME - 85/10/03. 10.58.02. WPAFB, DISSPLA, SOFEPL VERSION 2.2

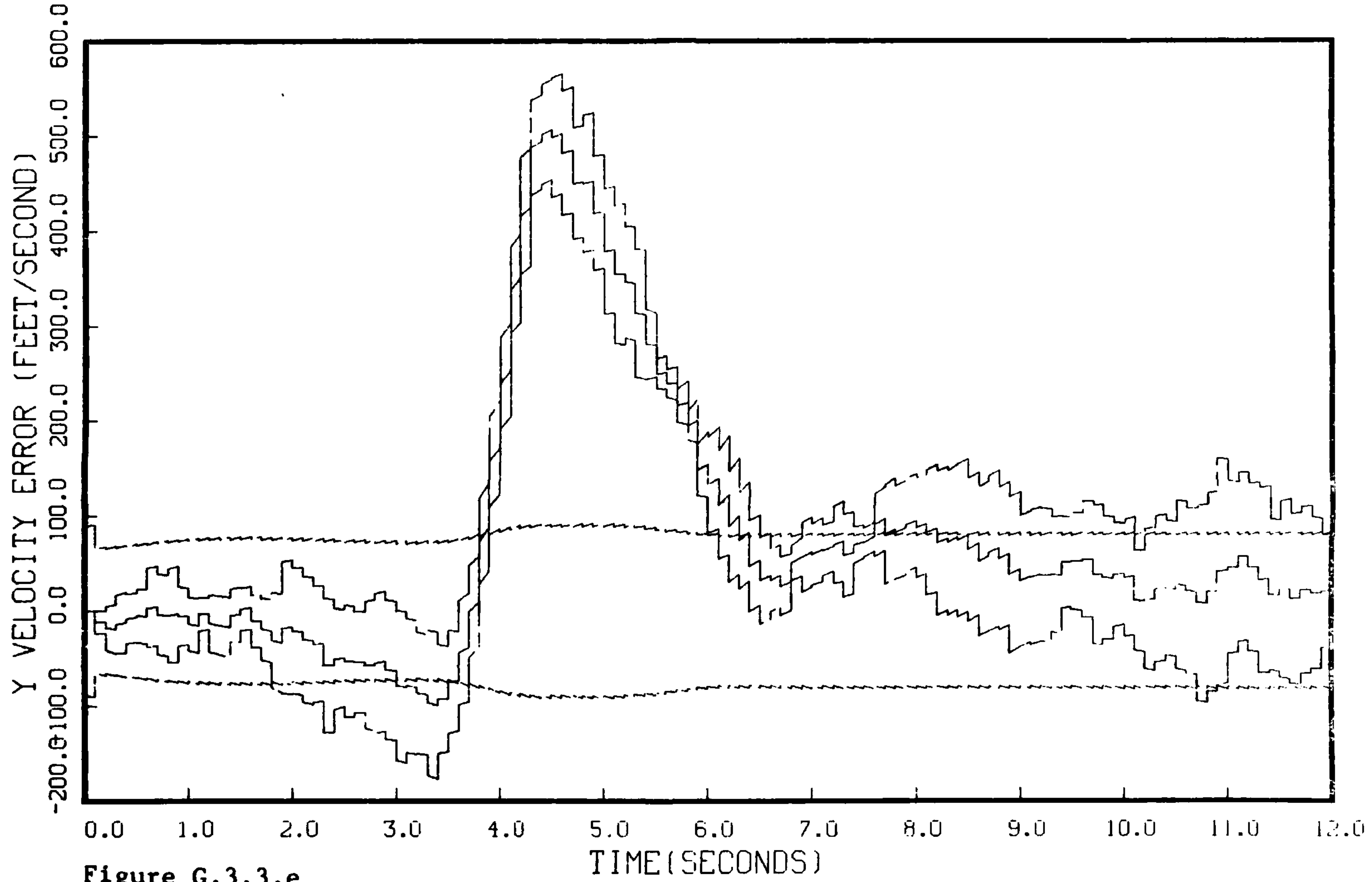


Figure G.3.3.e

STATE 5, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

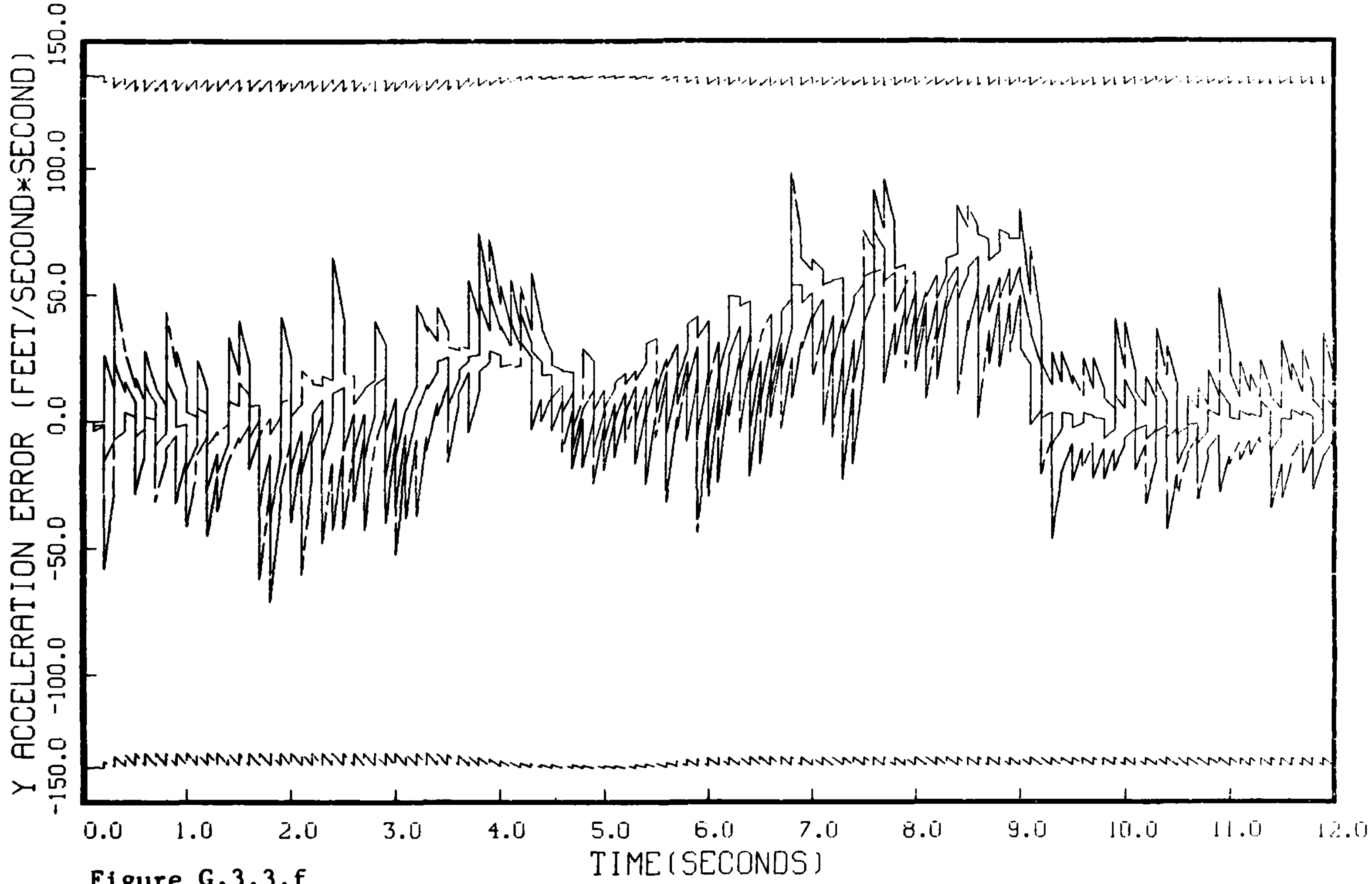


Figure G.3.3.f

STATE 6, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

7 SOFEPL DATE AND TIME - 85/10/03. 11.44.07. SOFE DATE AND TIME - 85/10/03. 10.58.02. WPAFB, DISSPLA, SOFEPL VERSION 2.2

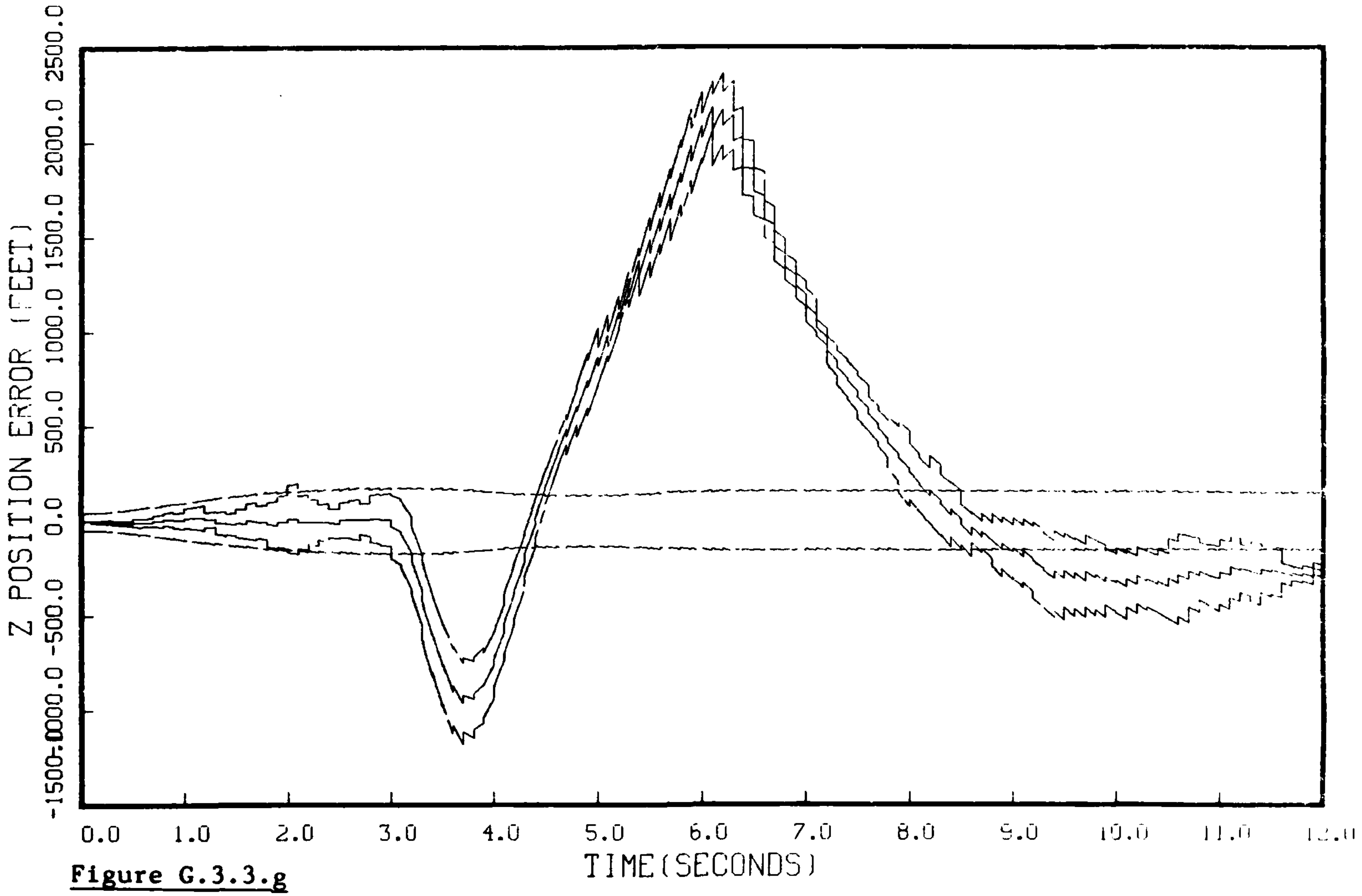


Figure G.3.3.g

STATE 7, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.1, 5 RUNS

8 SOFEPL DATE AND TIME - 85/10/03. 11.44.07. SOFE DATE AND TIME - 85/10/03. 10.58.02. WPAFB, DISSPLA, SOFEPL VERSION 2.2

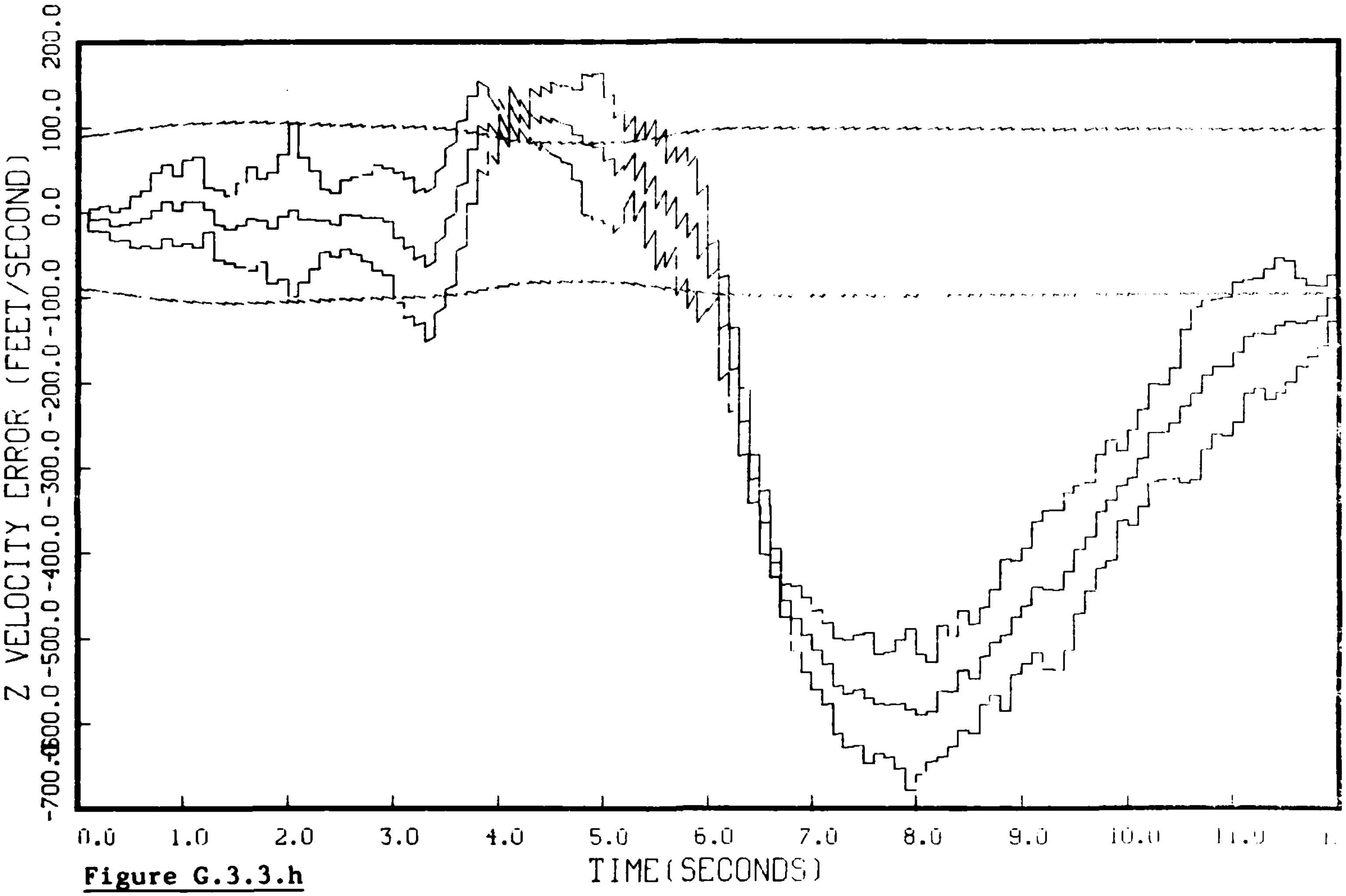


Figure G.3.3.h

STATE 8, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

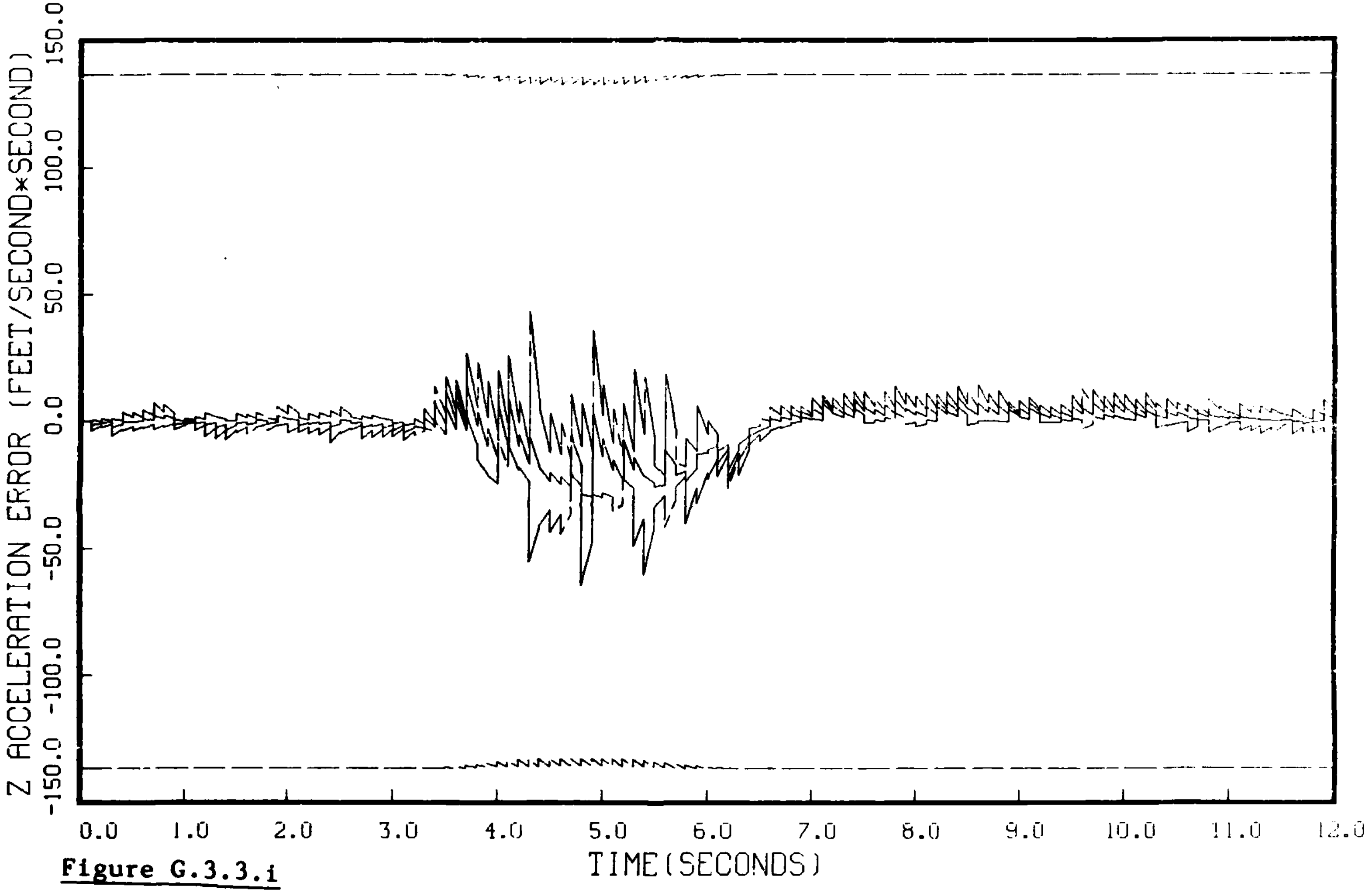
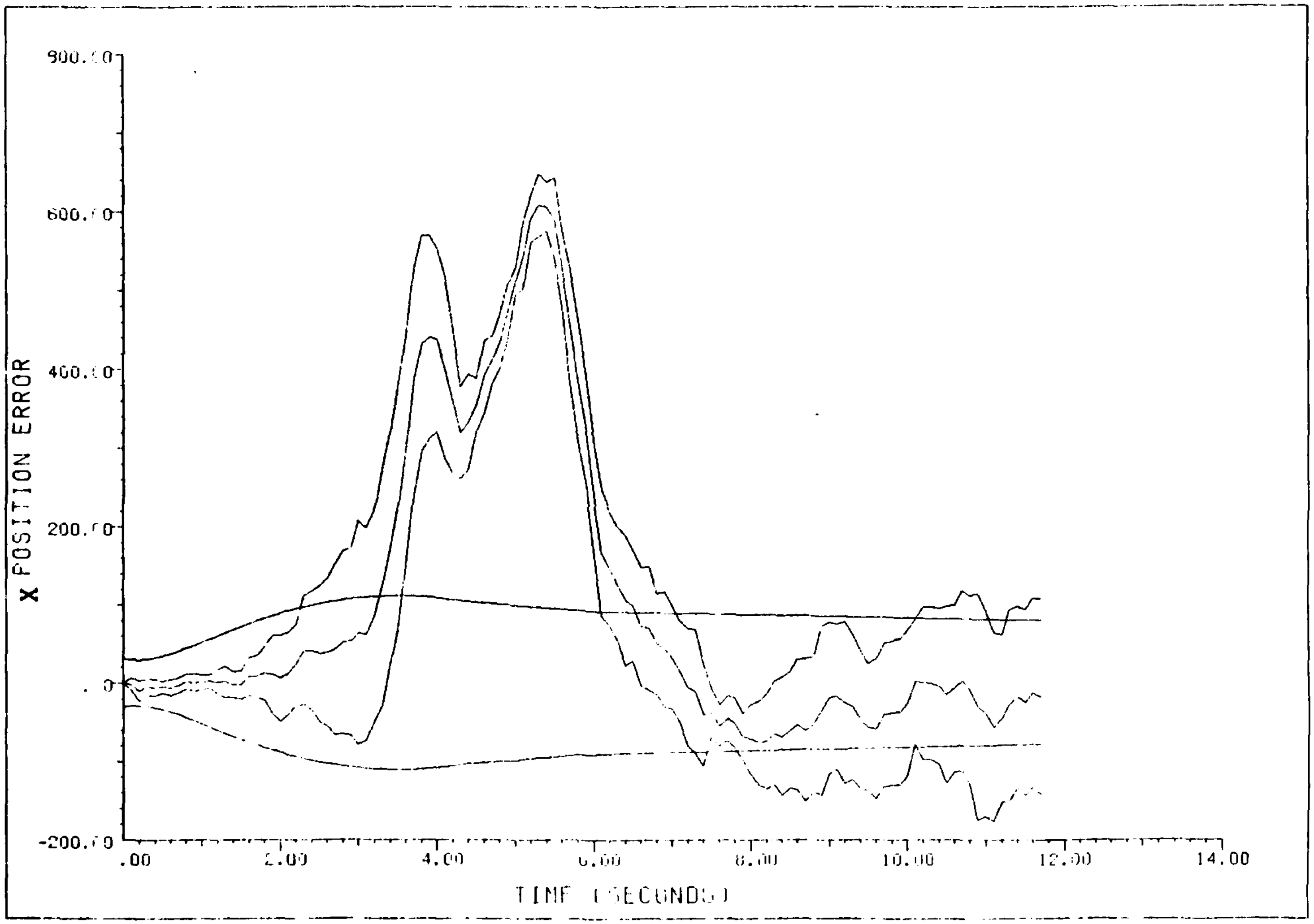


Figure G.3.3.i

STATE 9, Q(1)-Q(2)-Q(3)-261275., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

Figure Set G.3.4 is the same as Figure Set G.2.4

PSU #1



AFC120 Q =14390. TAU=.143, 6 MEAS. 5 RUNS

Figure G.4.1.a

APO120 Q=149300, TAU=.143, 5RUNS, 6 MEAS

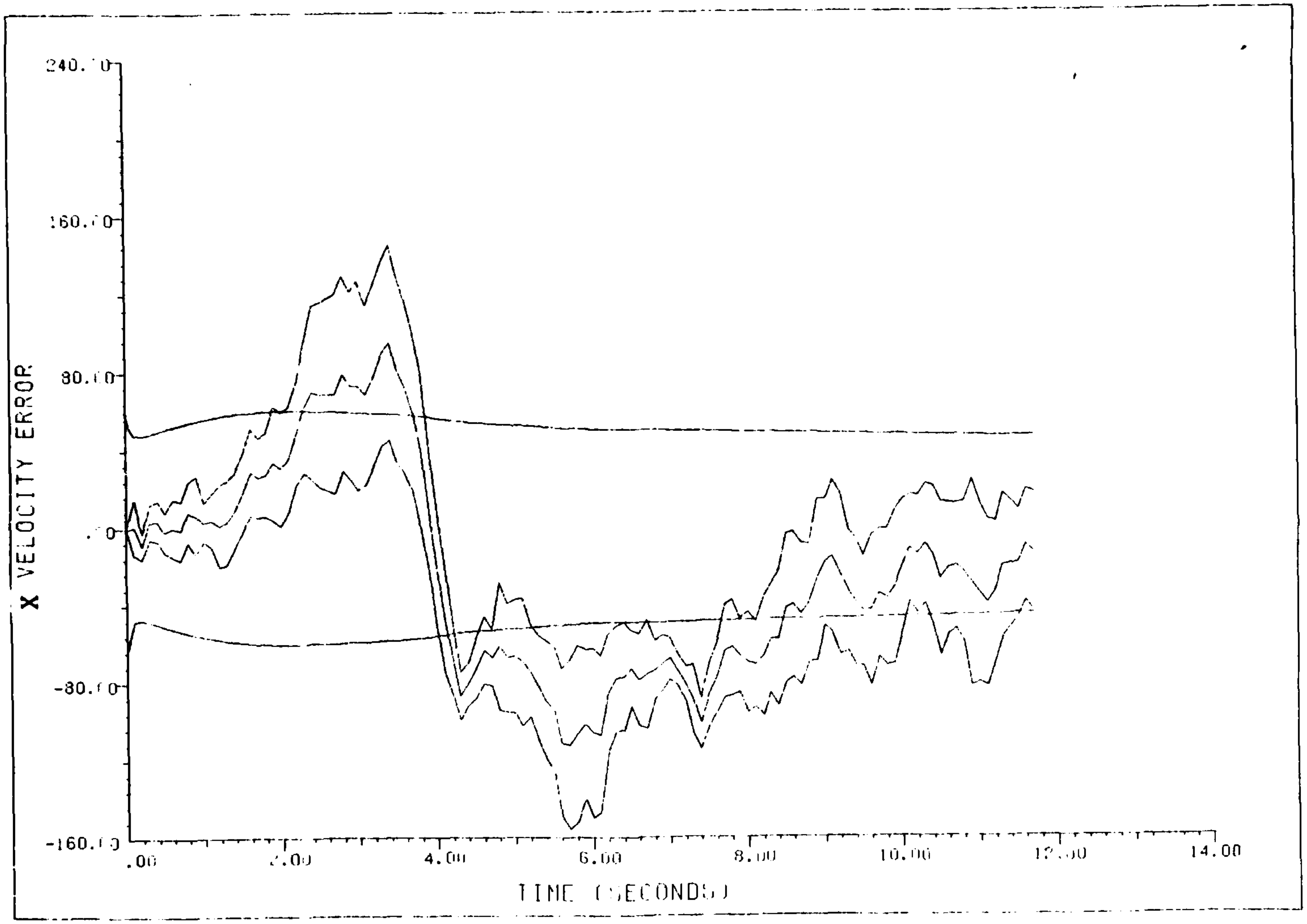
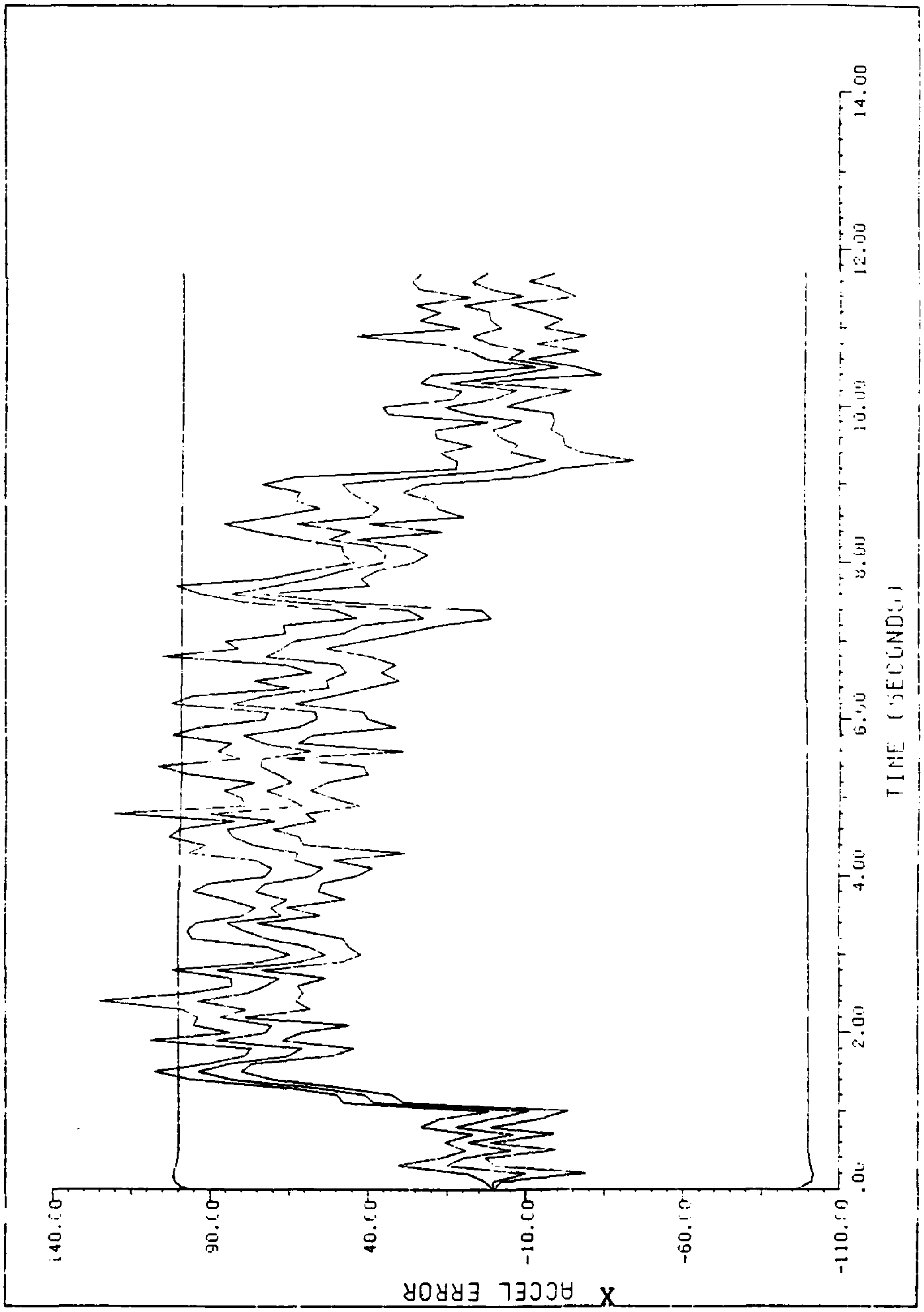
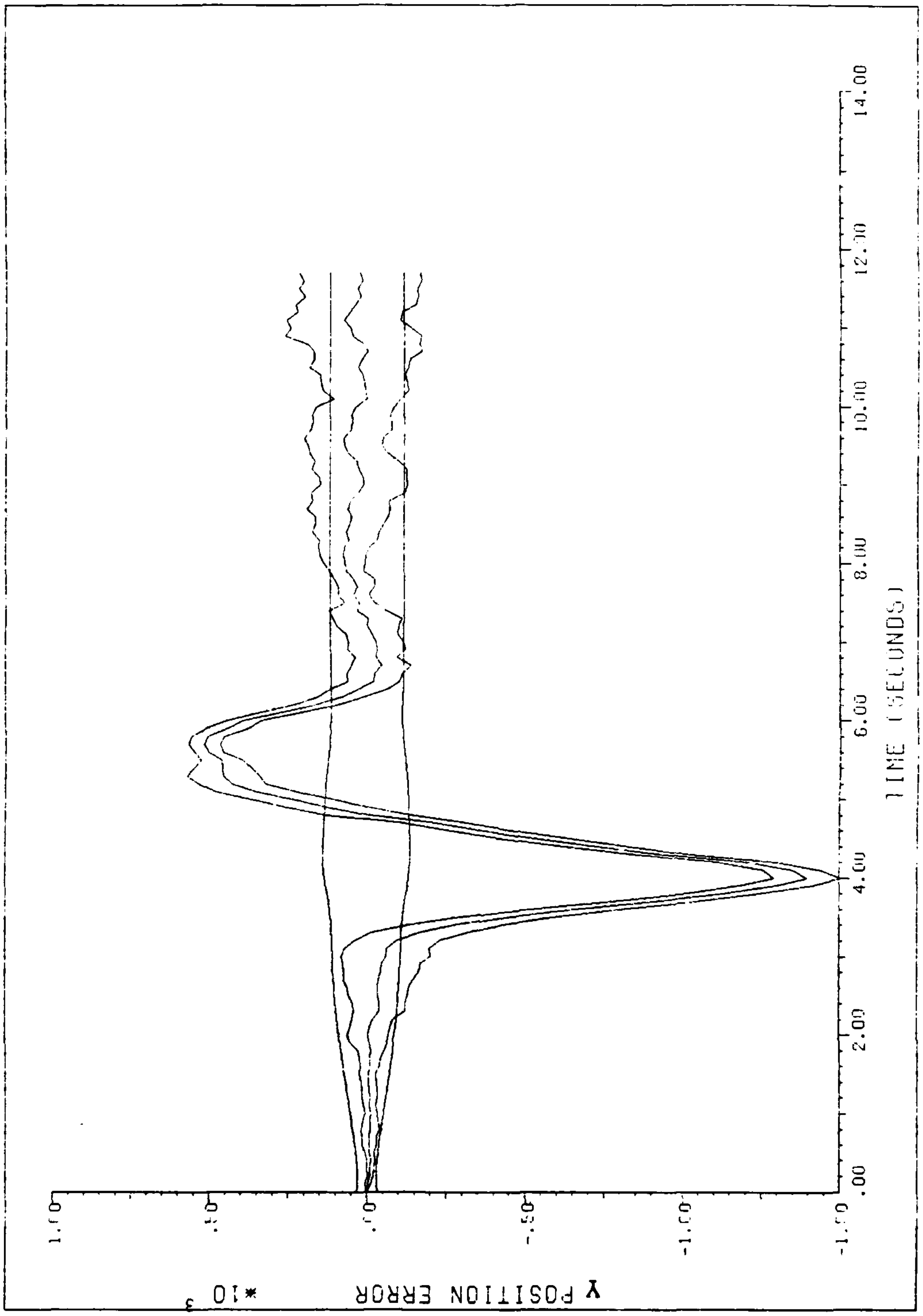


Figure G.4.1.b



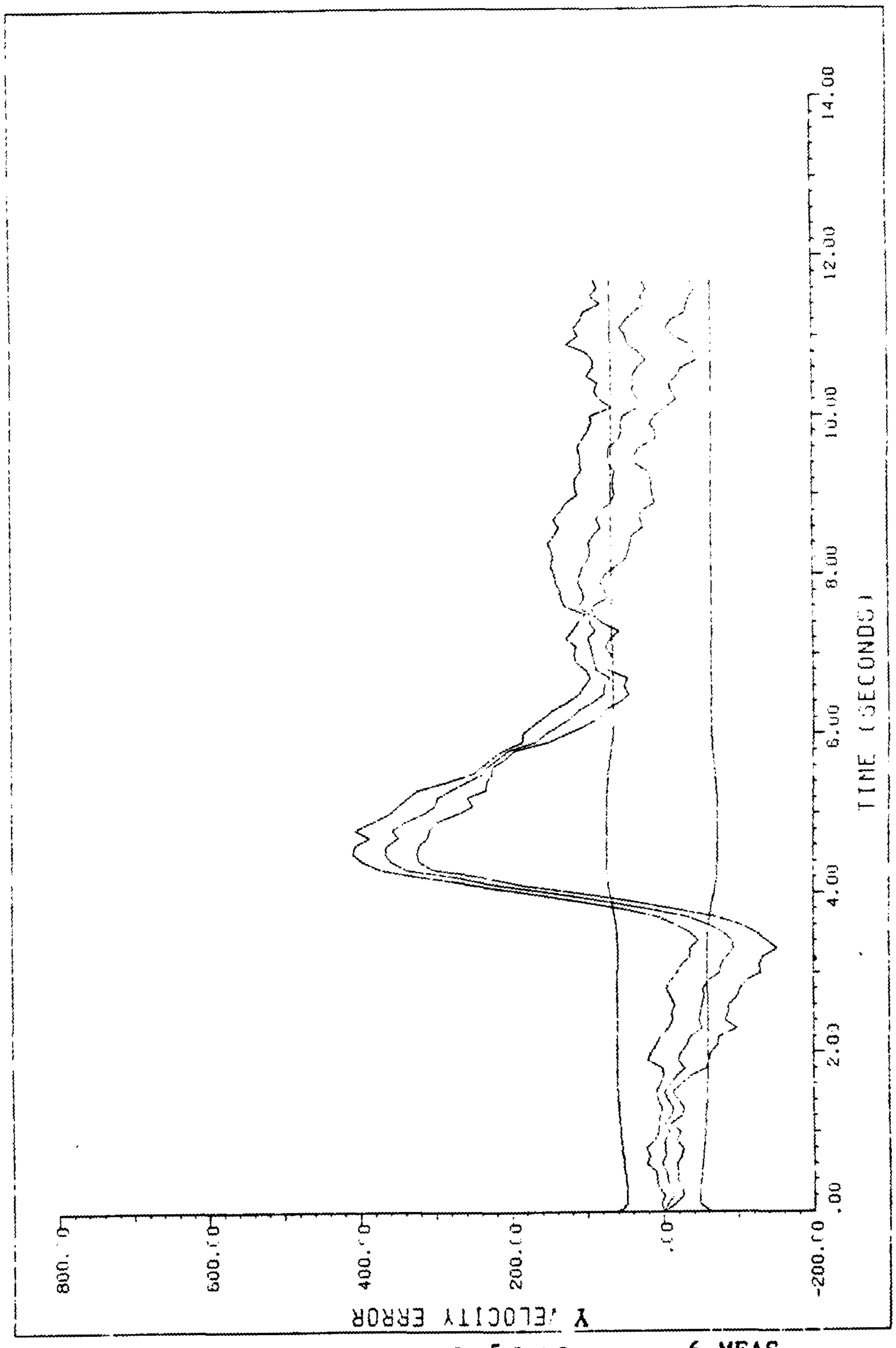
APQ120 Q=149300, TAU=.143, 5 RUNS, 6MEAS

Figure G.4.1.c



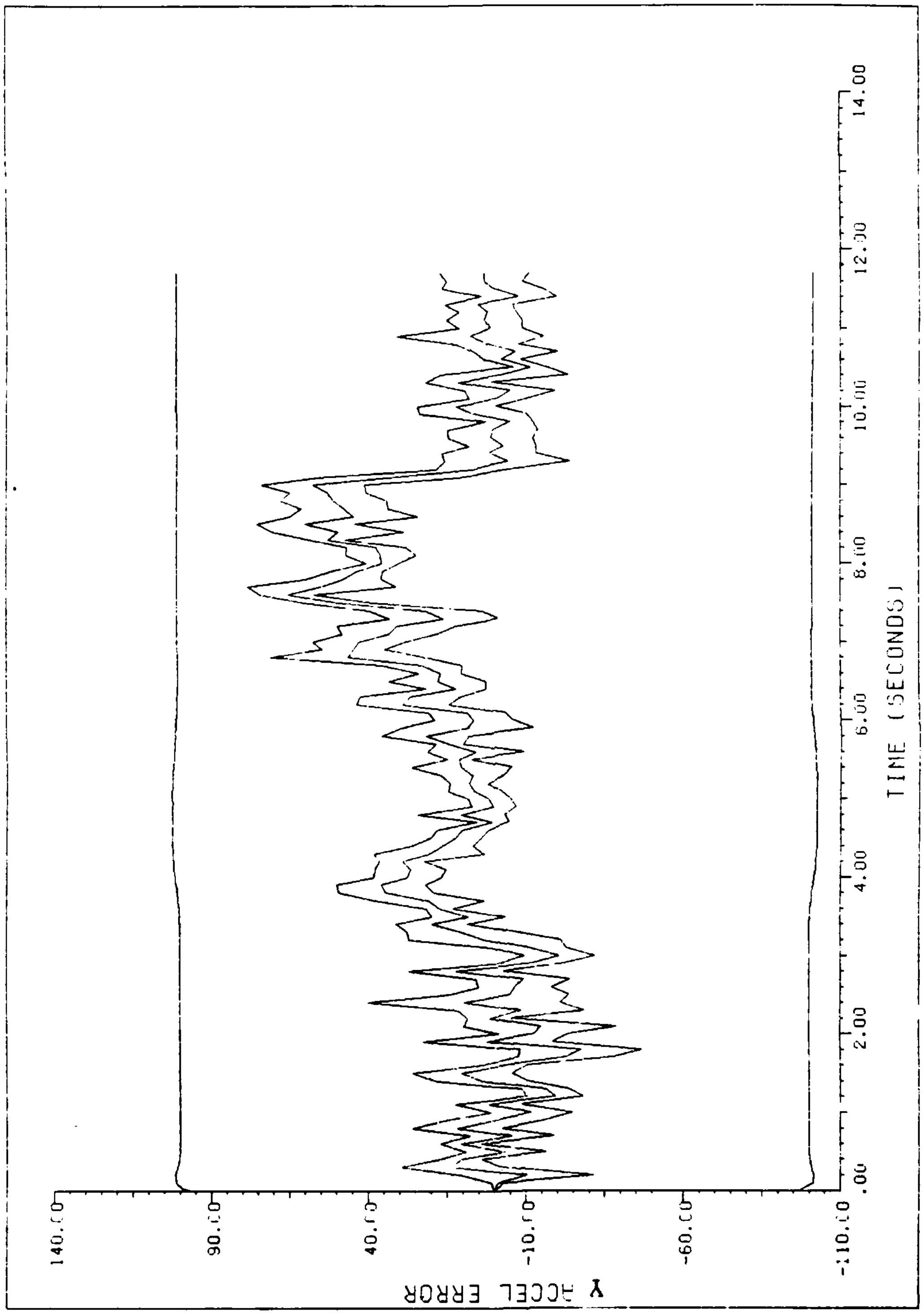
APQ120 Q =14390. TAU=.143. 6 MEAS. 5 RUNS

Figure G.4.1.d



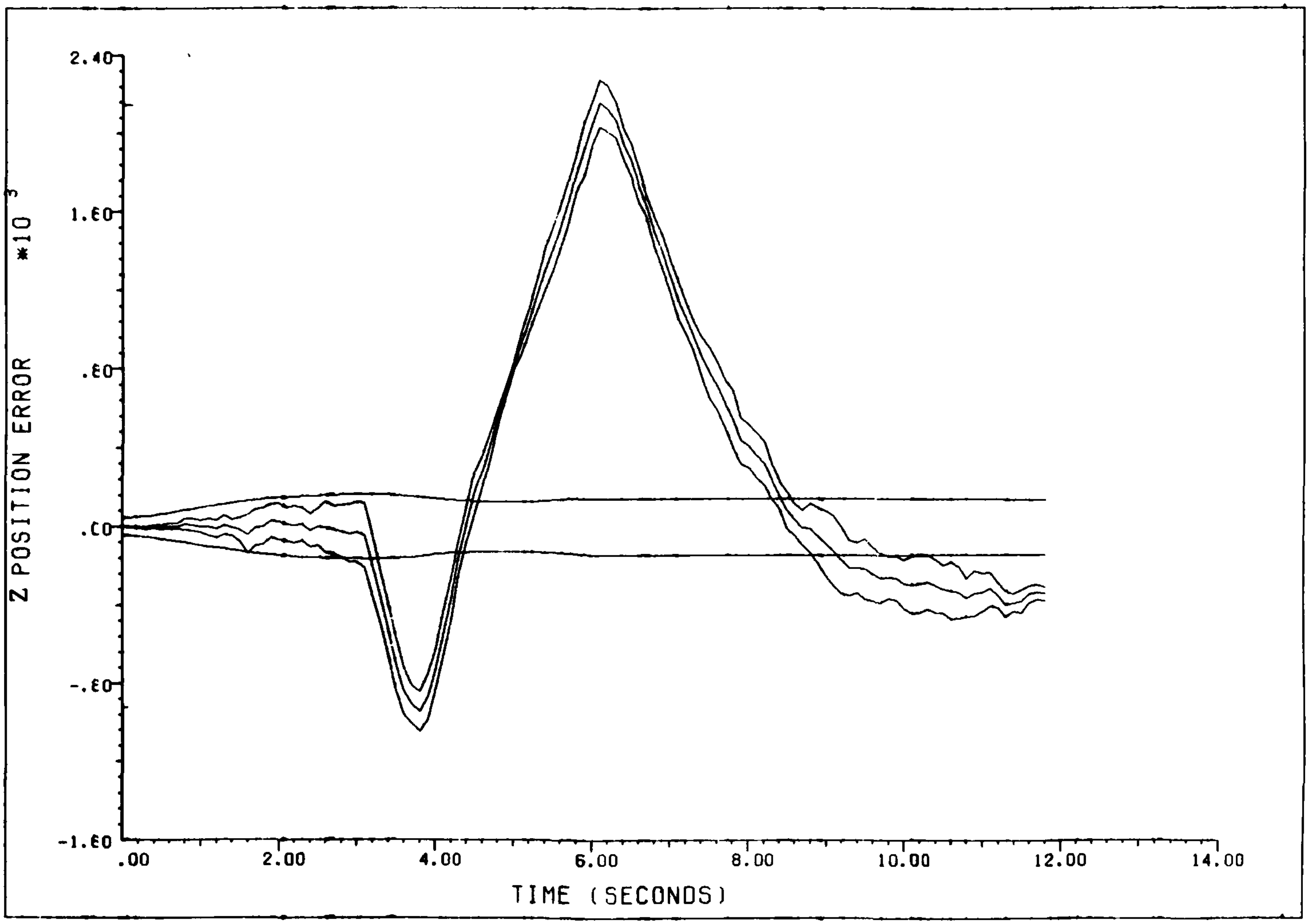
APQ120 Q=149300. TAU=.143. 5 RUNS. 6 MEAS

Figure G.4.1.e



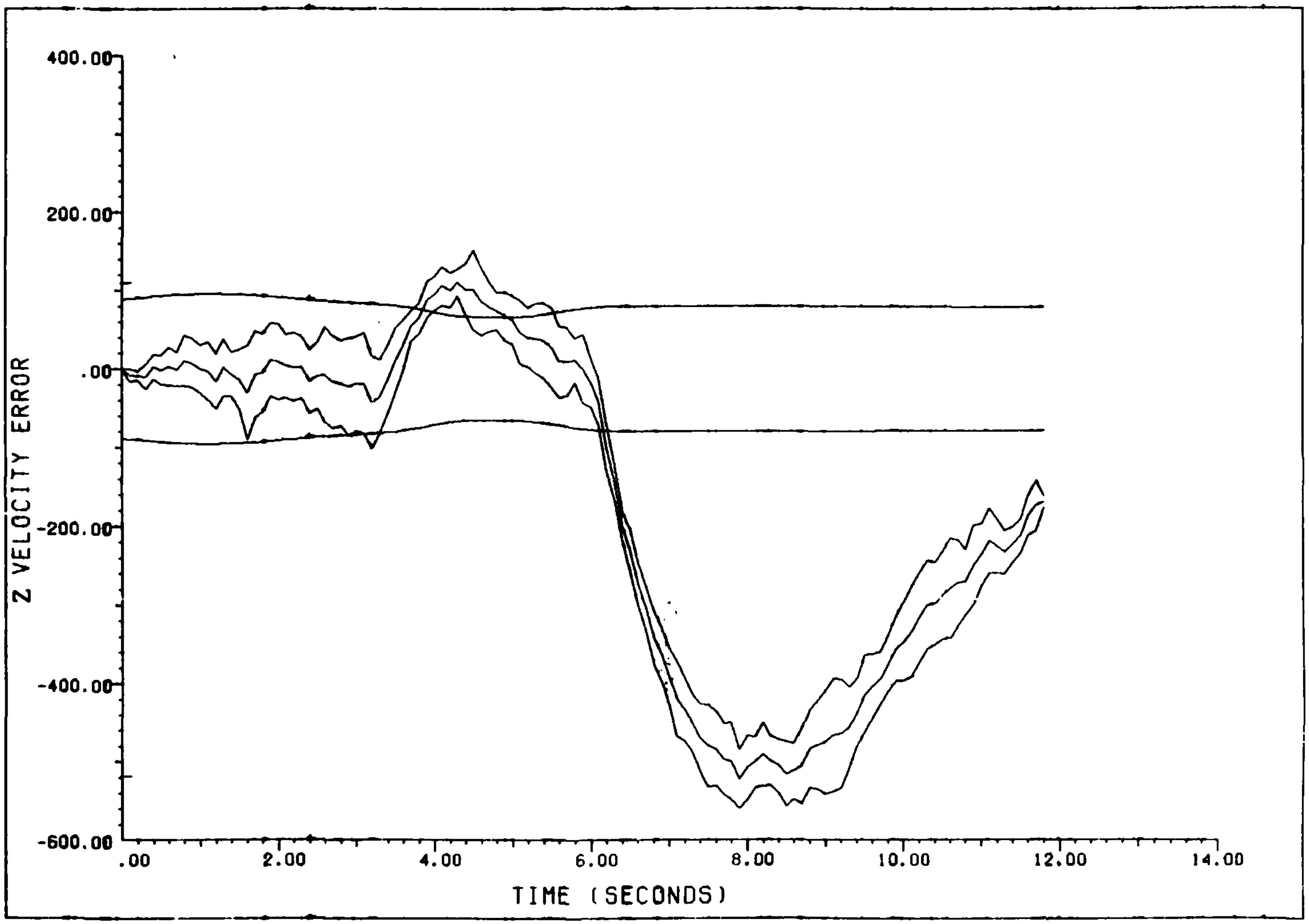
APQ120 Q=149300, TAU=.143, 5 RUNS, 6 MEAS

Figure G.4.1.f



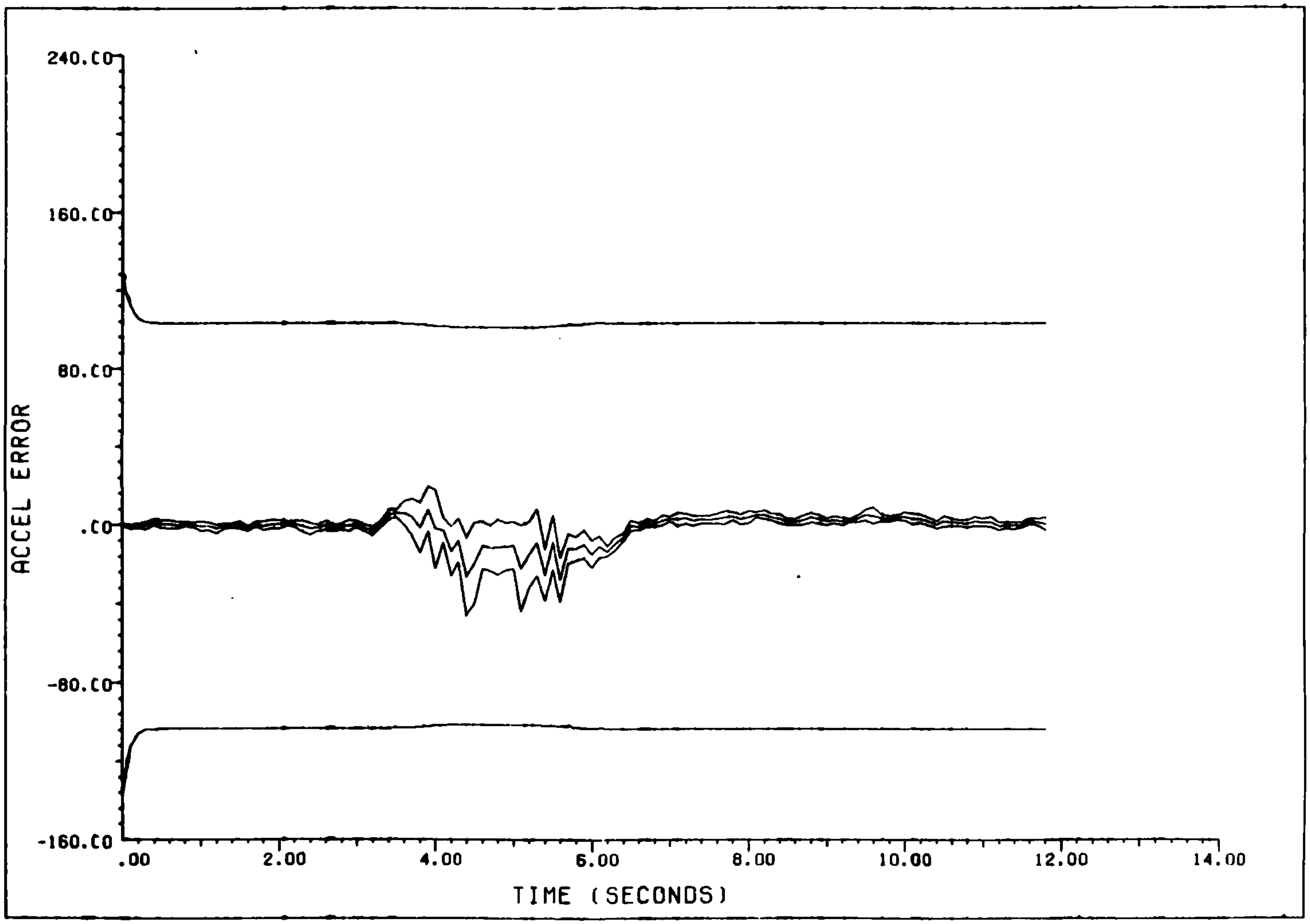
APQ120 Q =149300 TAU=.143 5 RUNS 6 MEAS

Figure G.4.1.g



APQ120 Q= 149300 TAU= .143 5 RUNS 6 MEAS

Figure G.4.1.h



APQ120 Q= 149300 TAU= .143 5RUNS 6 MEAS

Figure G.4.1.i

Figure Set G.4.2 is the same as Figure Set G.3.2

73 P54 3 1 SOFEPL DATE AND TIME - 85/10/04. 07.30.38. SOFE DATE AND TIME - 85/10/03. 22.14.32. WPAFB, DISSPLA, SOFEPL VERSION 2.2

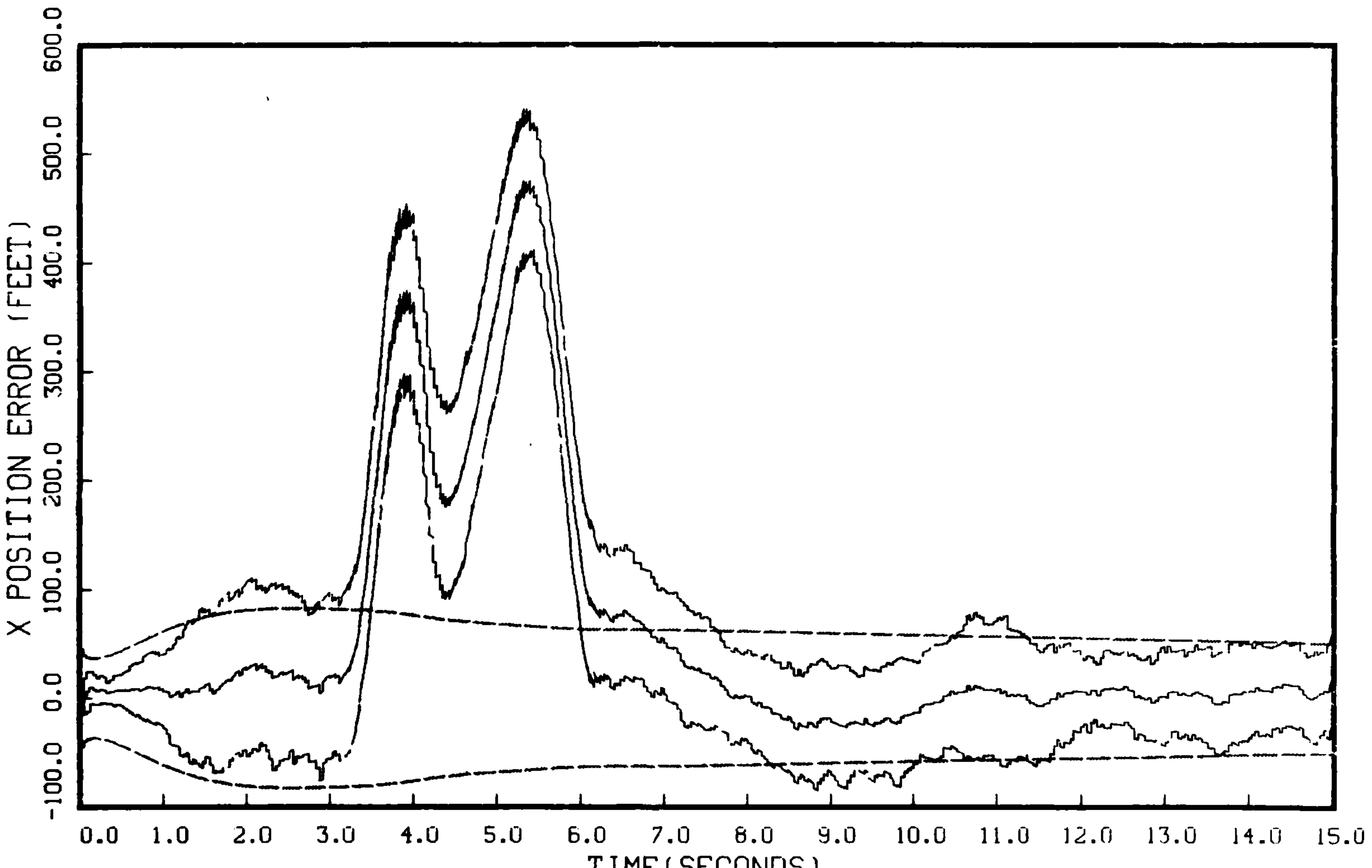


Figure G.4.3.a

STATE 1, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.04, 20 RUNS

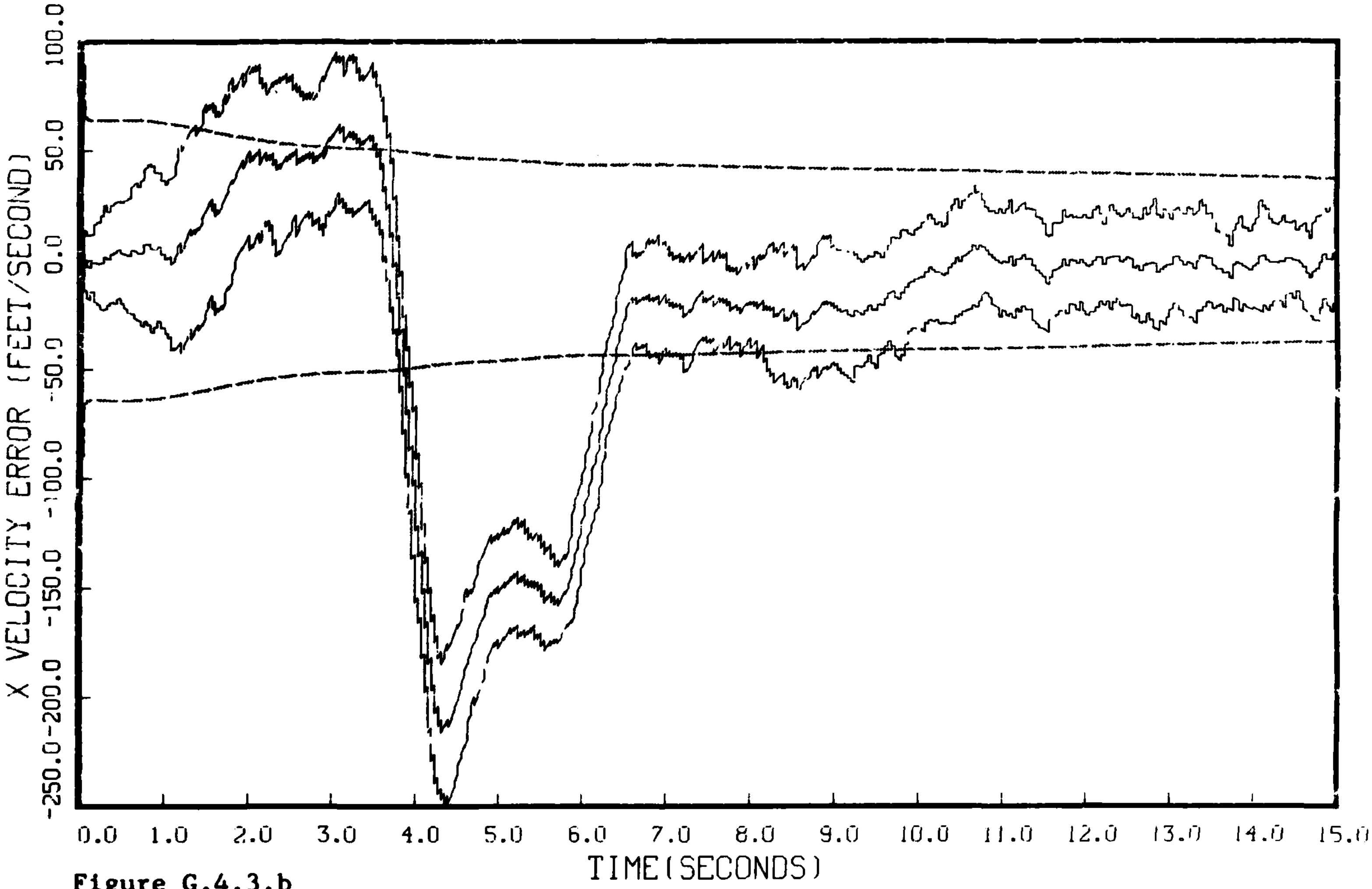


Figure G.4.3.b

STATE 2, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.04, 20 RUNS

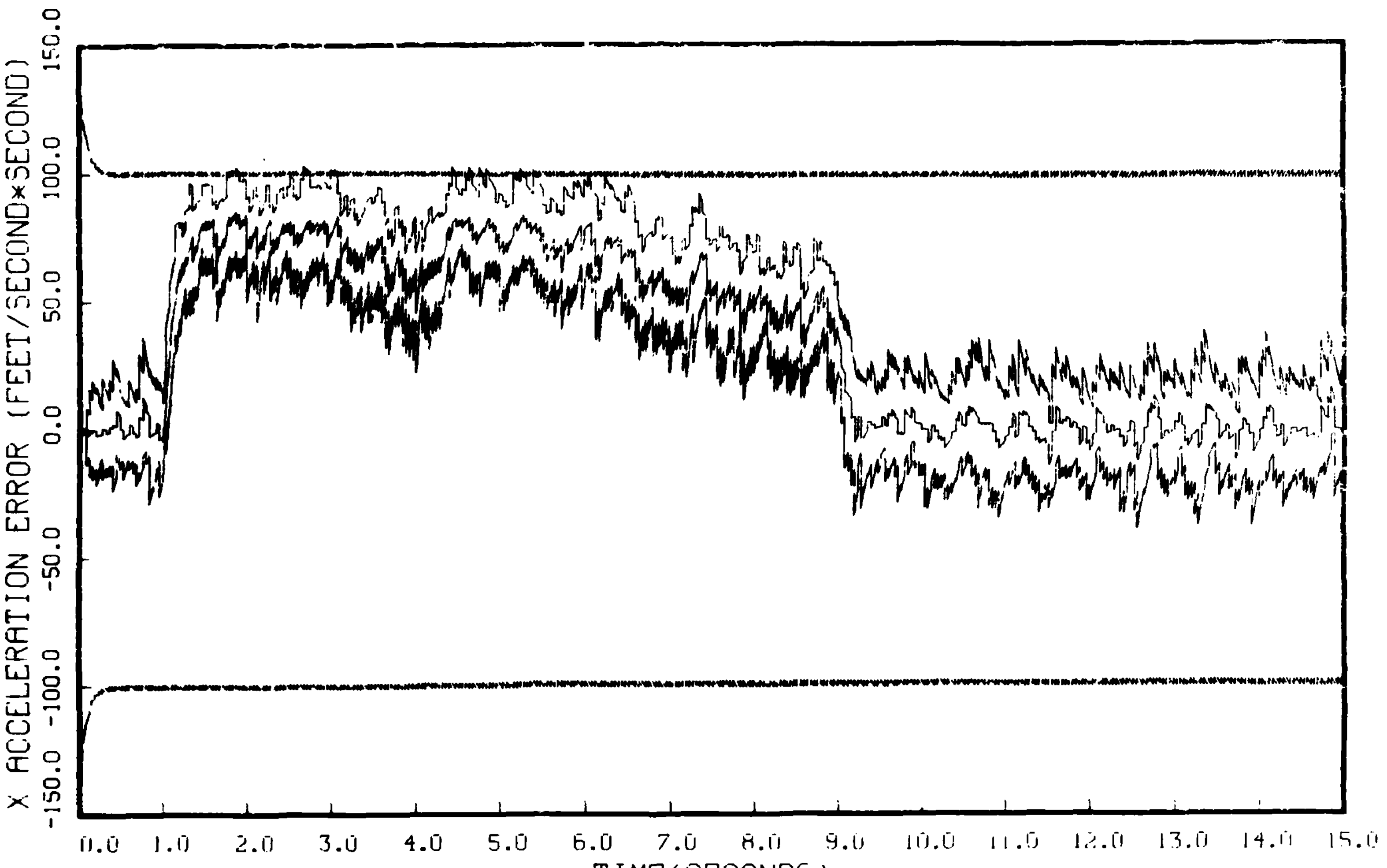


Figure G.4.3.c

STATE 3, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.04, 20 RUNS

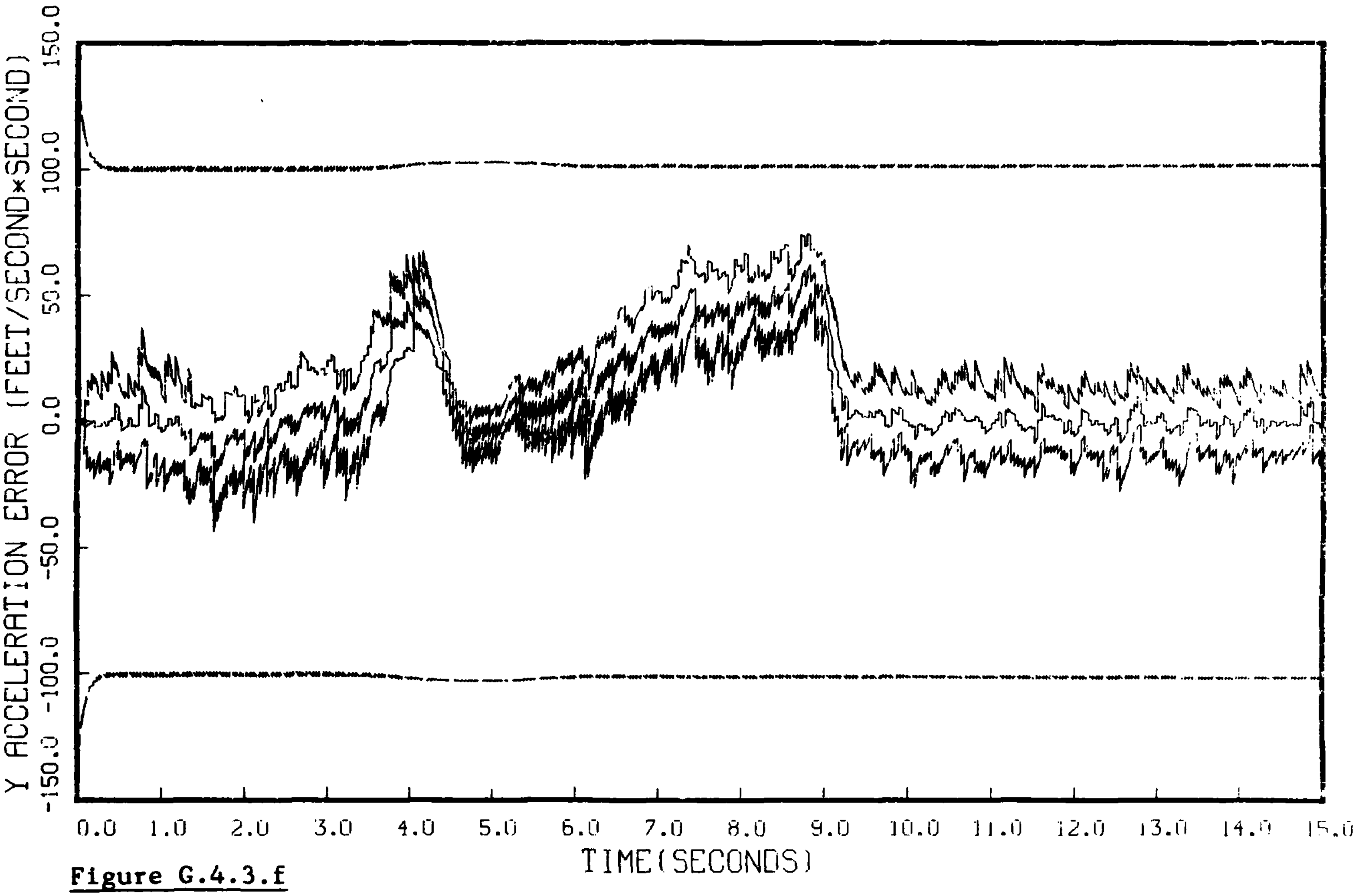


Figure G.4.3.f

STATE 6, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.04, 20 RUNS

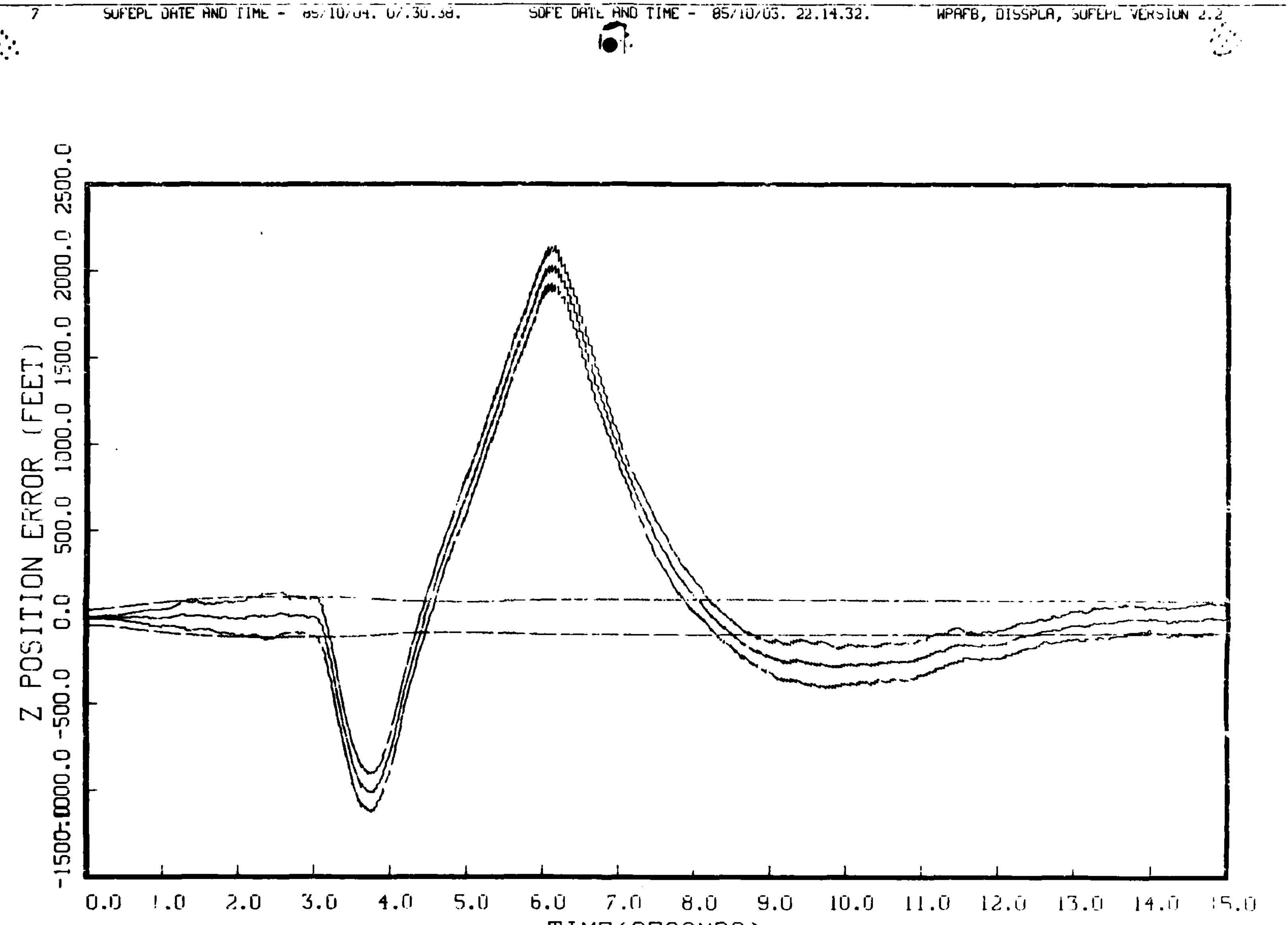


Figure G.4.3.g

STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, DCAM ATTACK, INITIAL RANGE-40,000. , UPDATE-.04, 20 RUNS

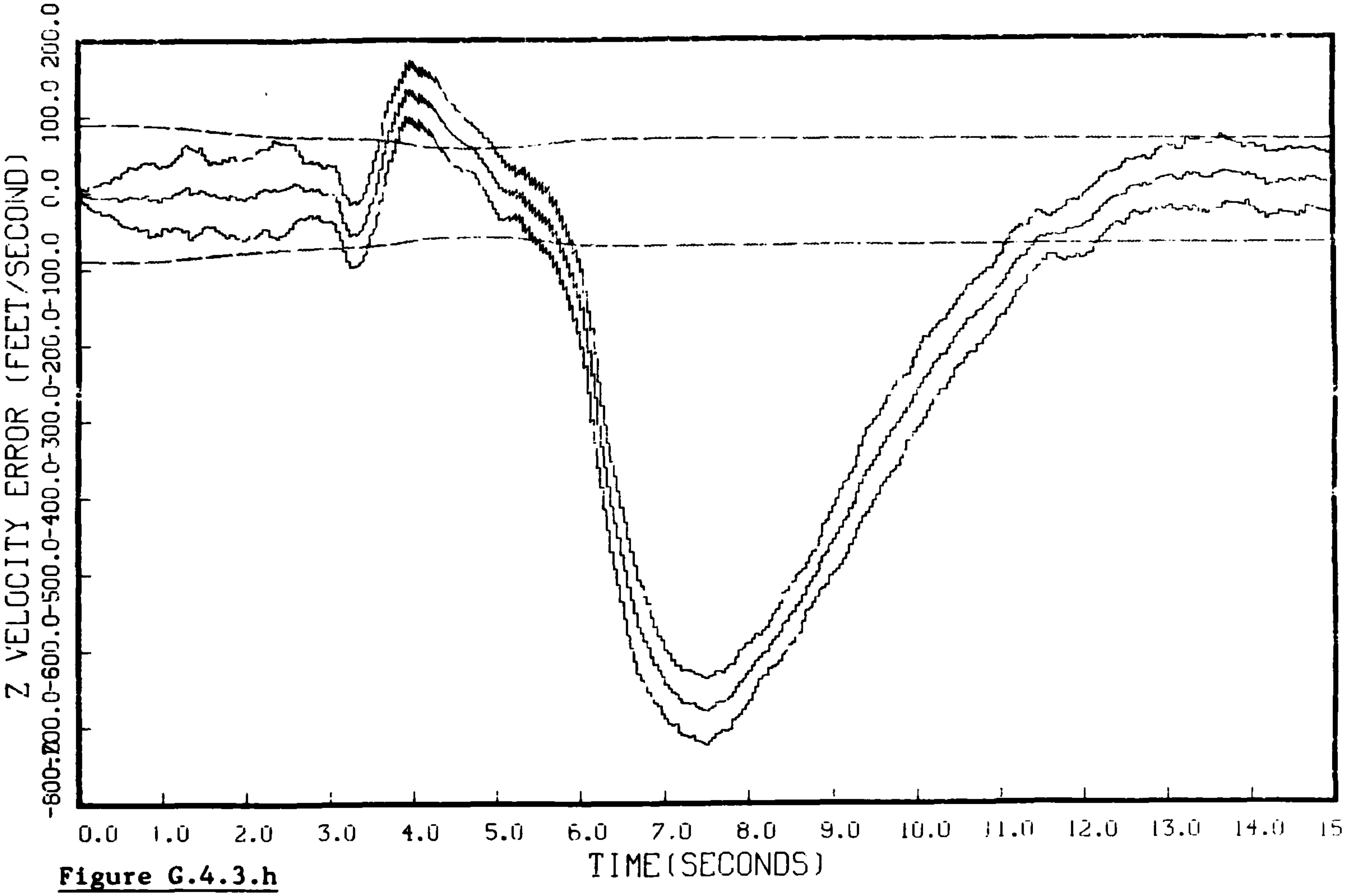


Figure G.4.3.h

STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.04, 20 RUNS

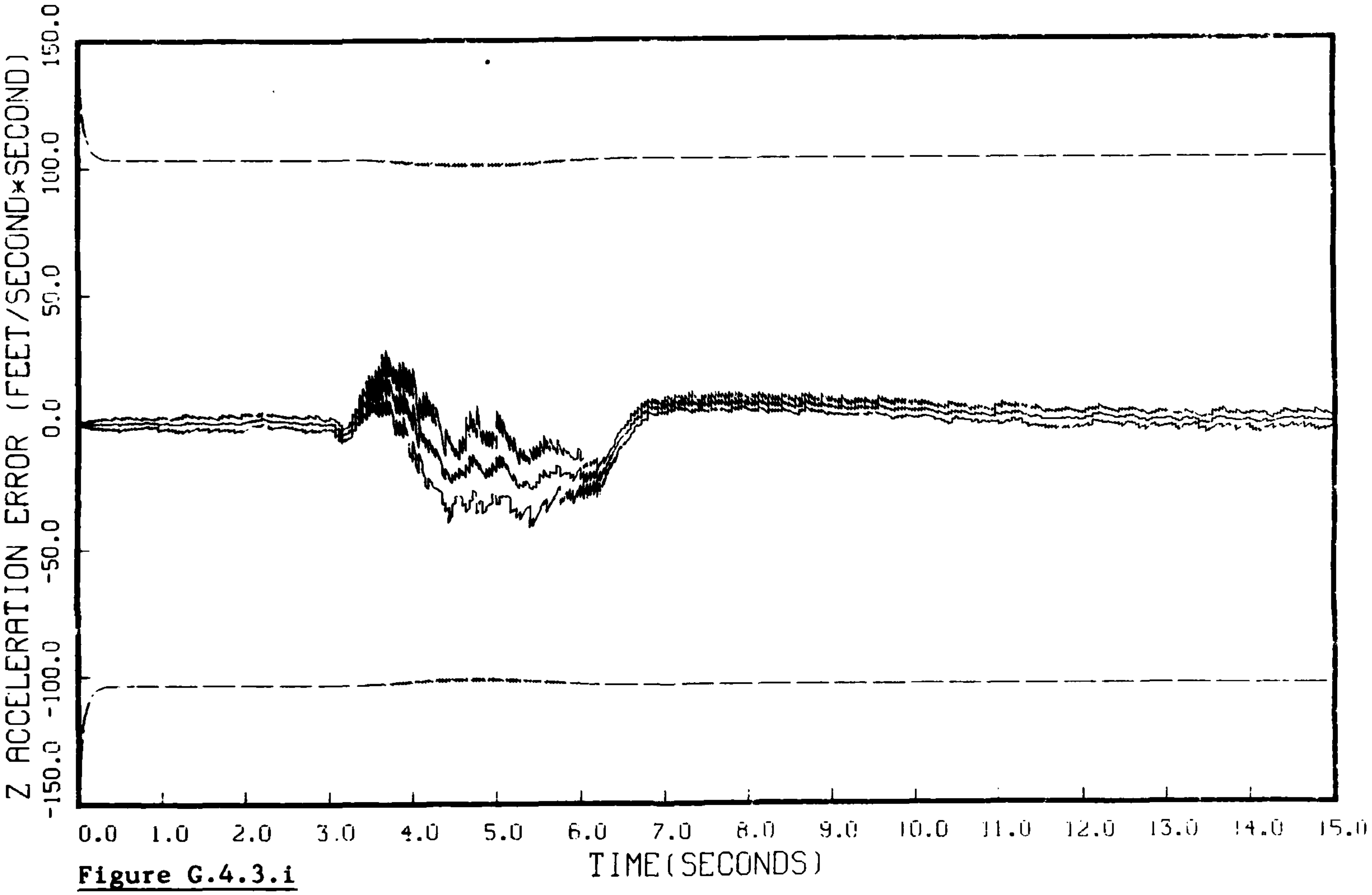


Figure G.4.3.i

STATE 9, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.04, 20 RUNS

7A PS4 4 1 SOFEPL DATE AND TIME - 85/10/12. 14.38.41. SOFE DATE AND TIME - 85/10/12. 14.31.10. WPAFB, DISSPLA, SOFEPL VERSION 2.2

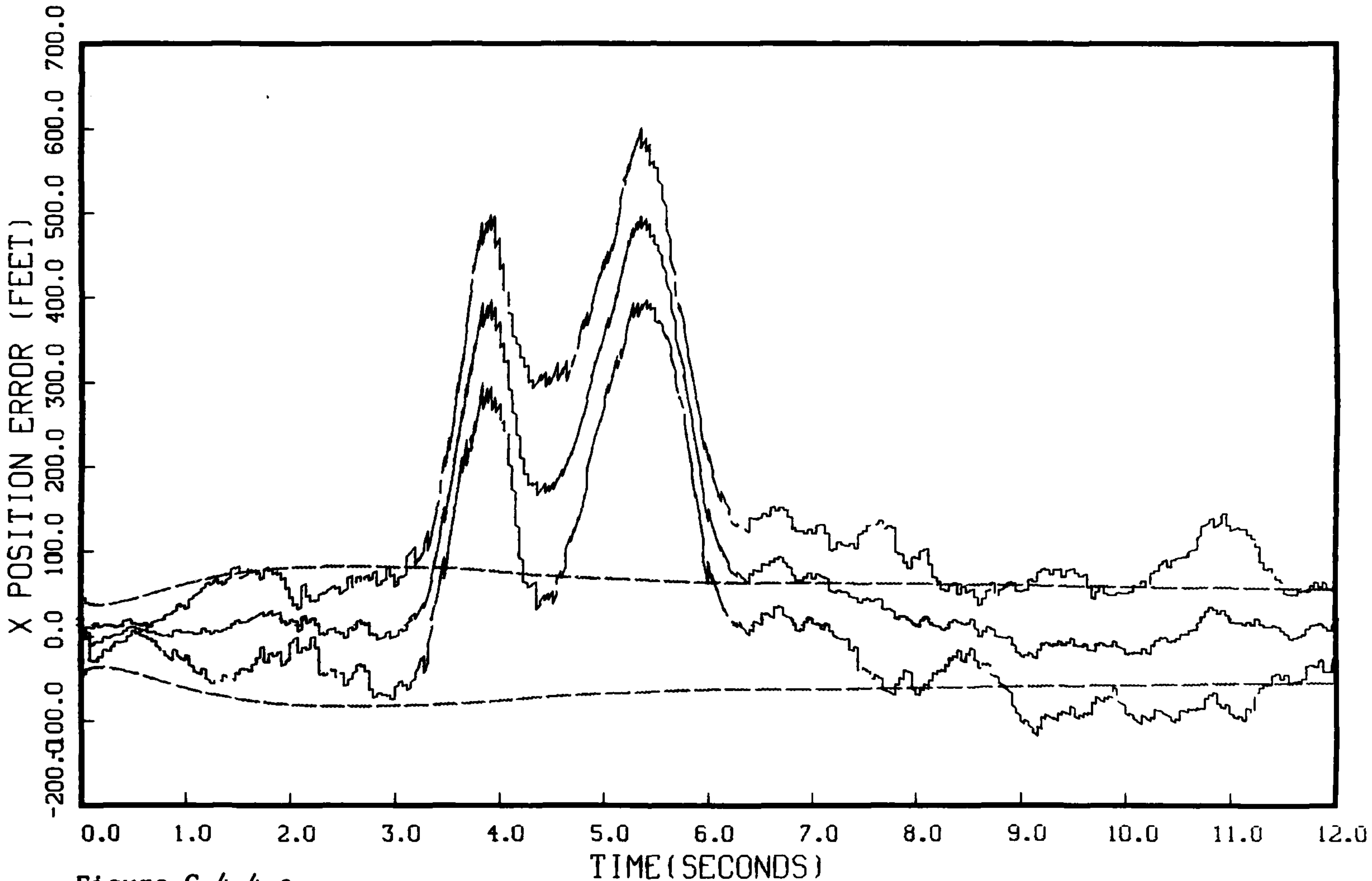


Figure G.4.4.a

STATE 1, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.04, 5 RUNS

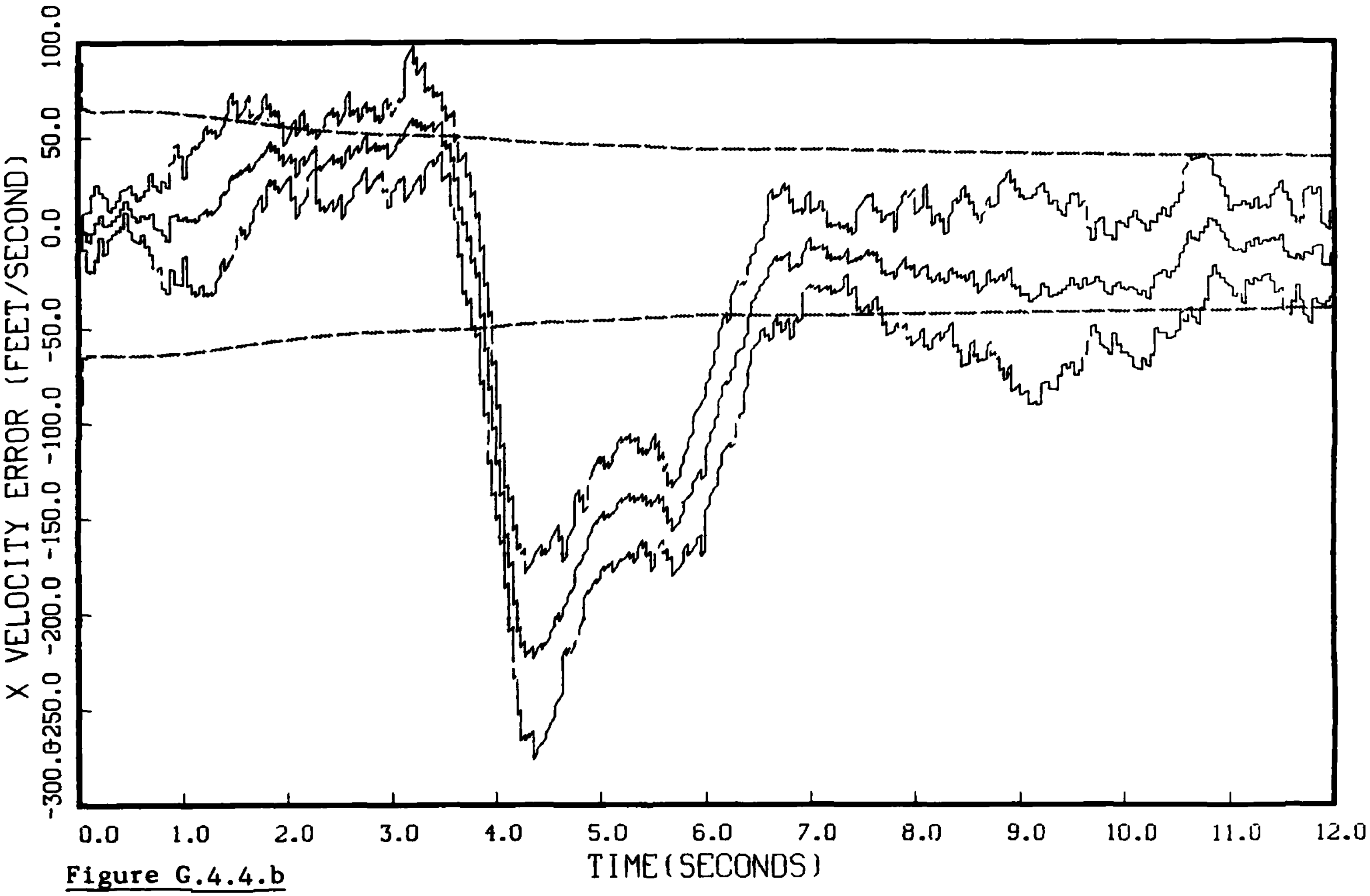


Figure G.4.4.b

STATE 2, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.04, 5 RUNS

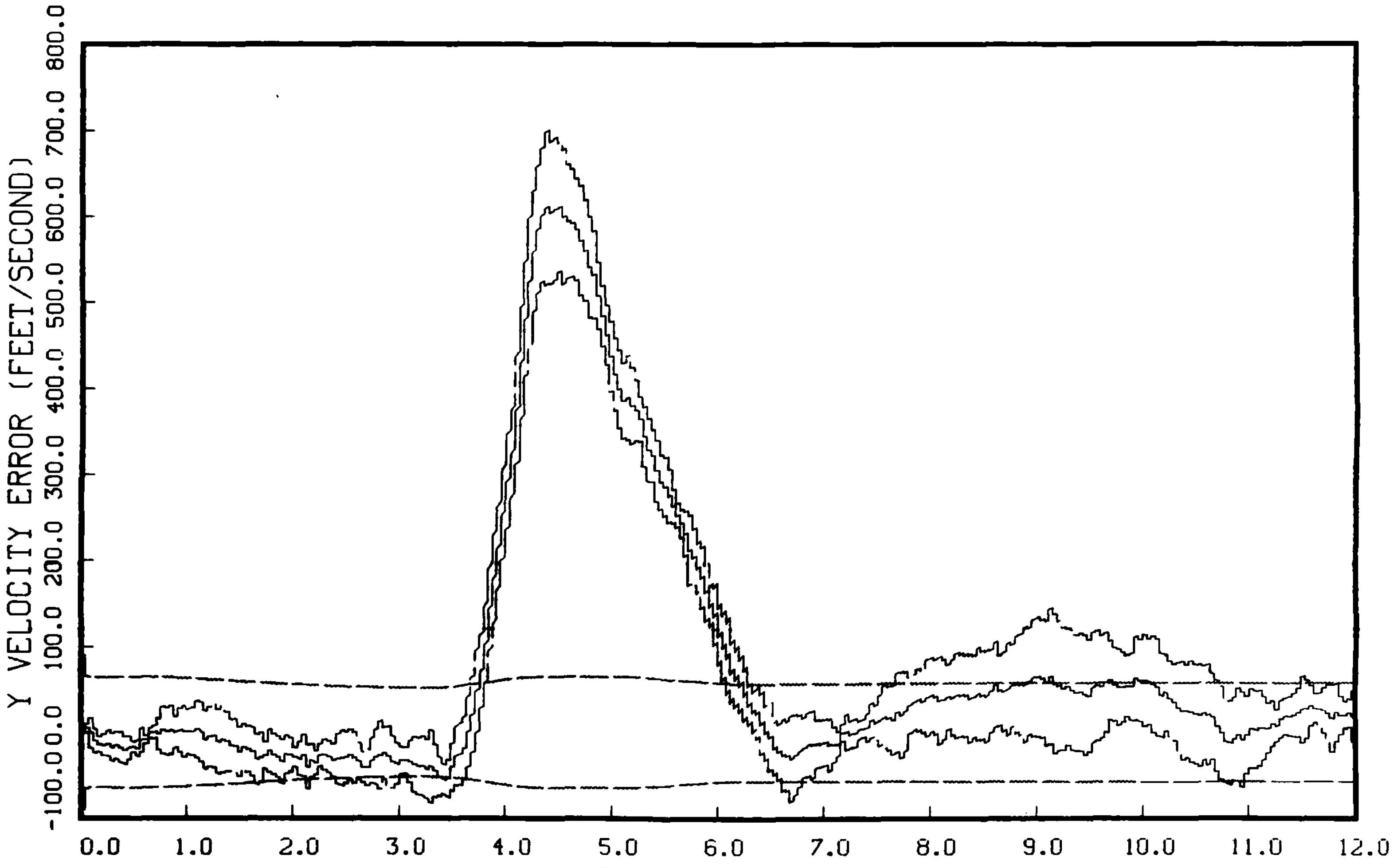


Figure G.4.4.e

STATE 5, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

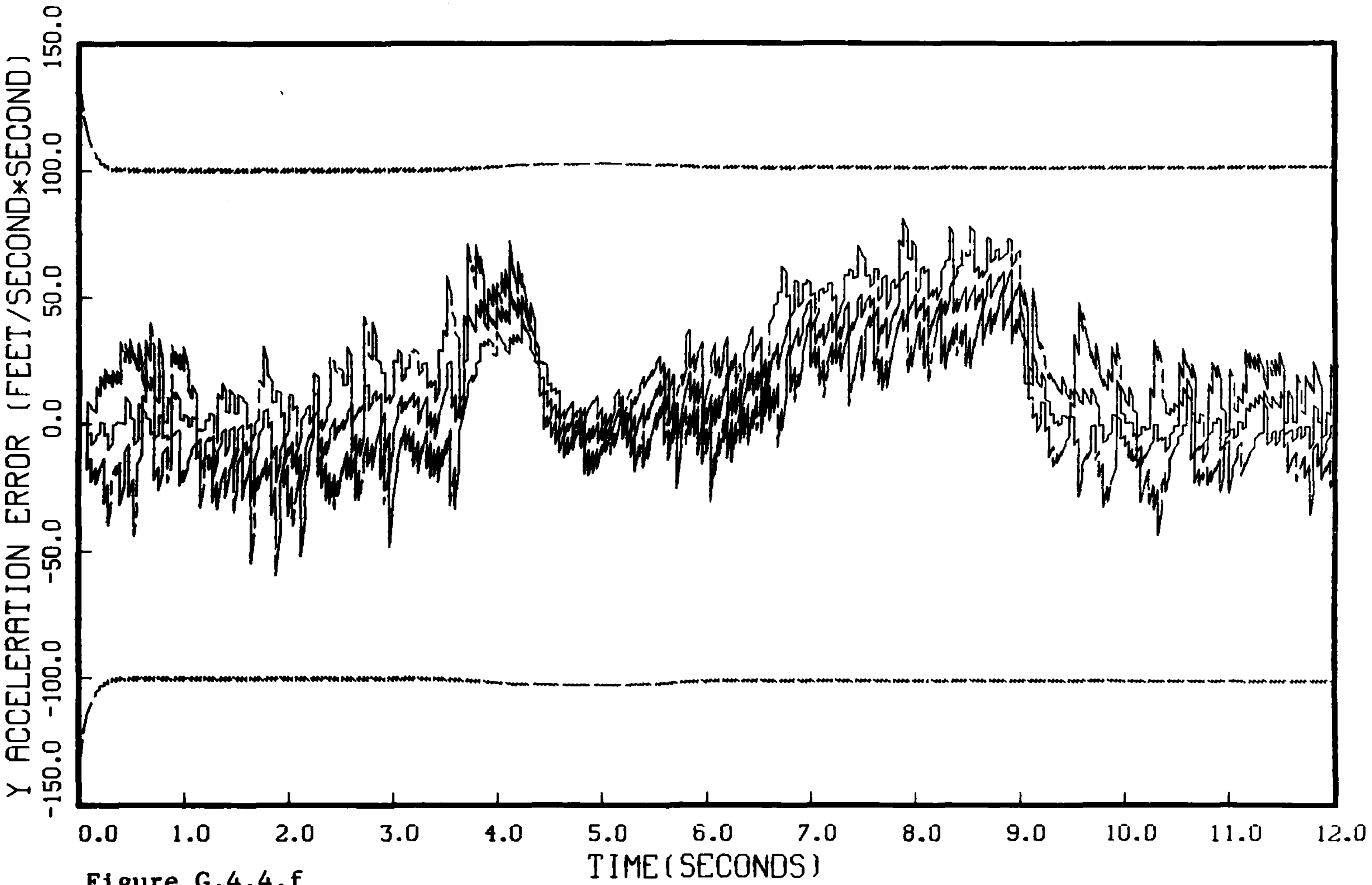


Figure G.4.4.f

STATE 6, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

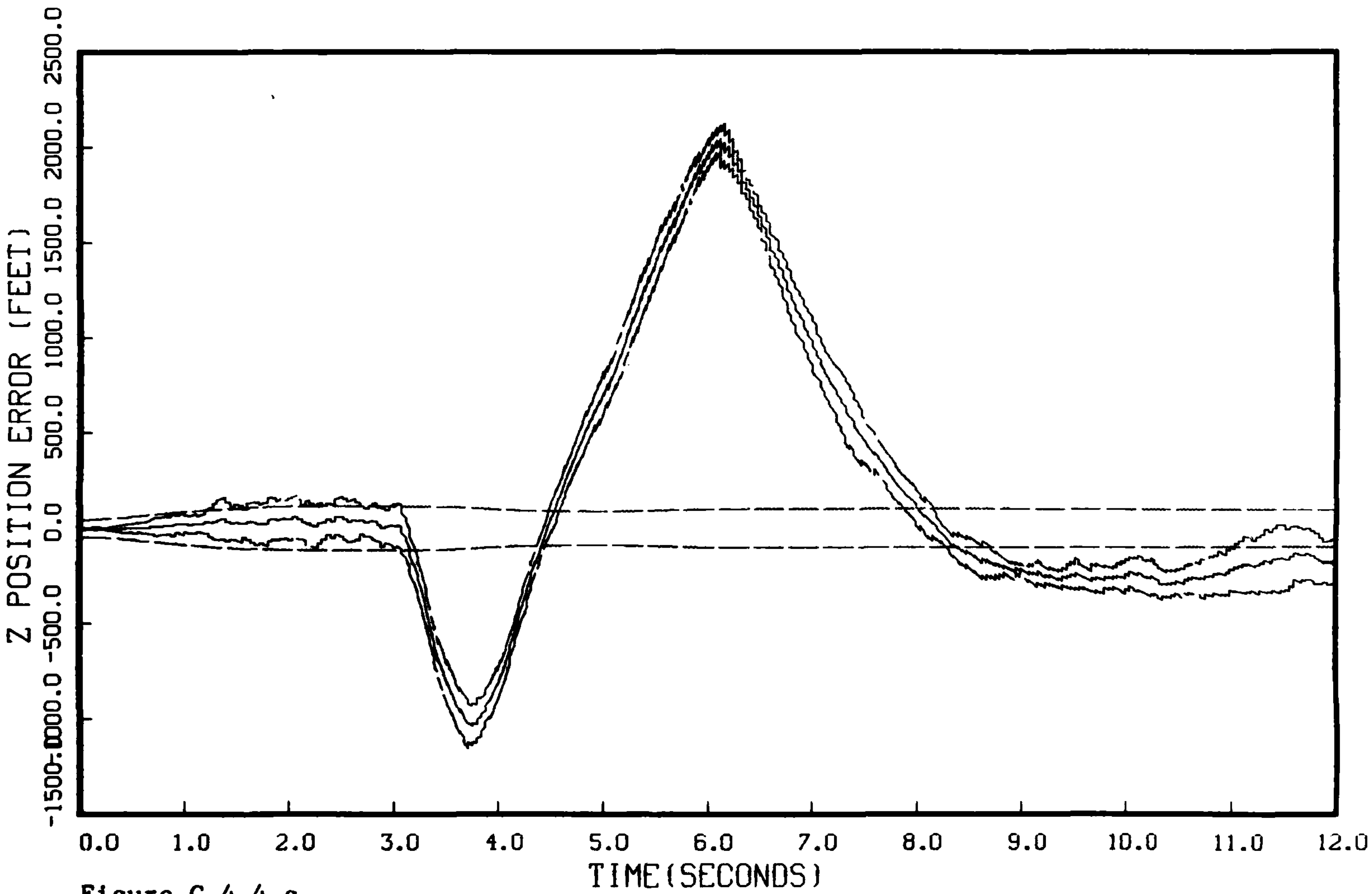


Figure G.4.4.g

STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

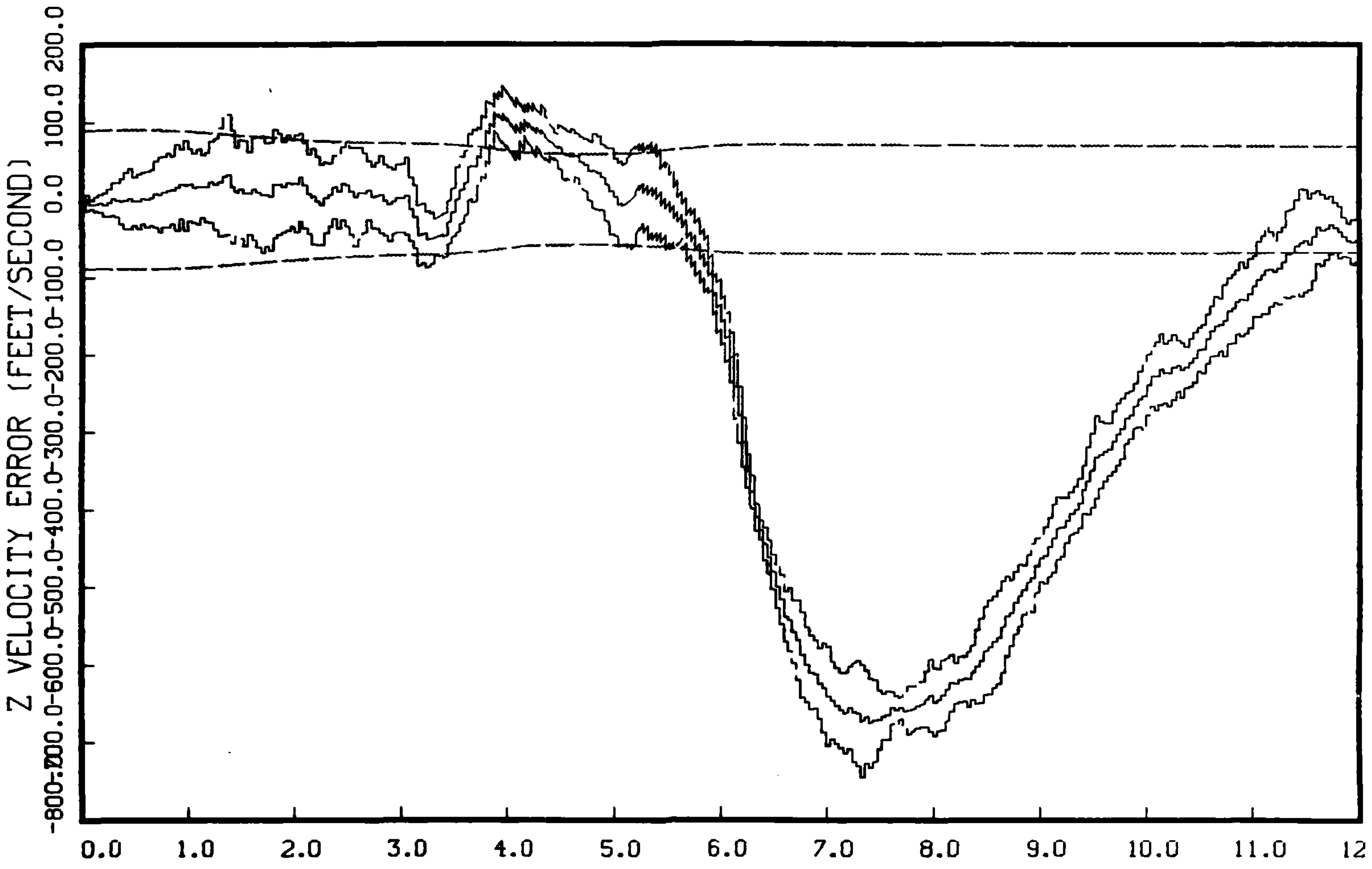


Figure G.4.4.h

STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

7 SOFEPL DATE AND TIME - 85/10/04. 21.24.31. SOFE DATE AND TIME - 85/10/04. 20.38.18. WPAFB, DISSPLA, SOFEPL VERSION 2.2

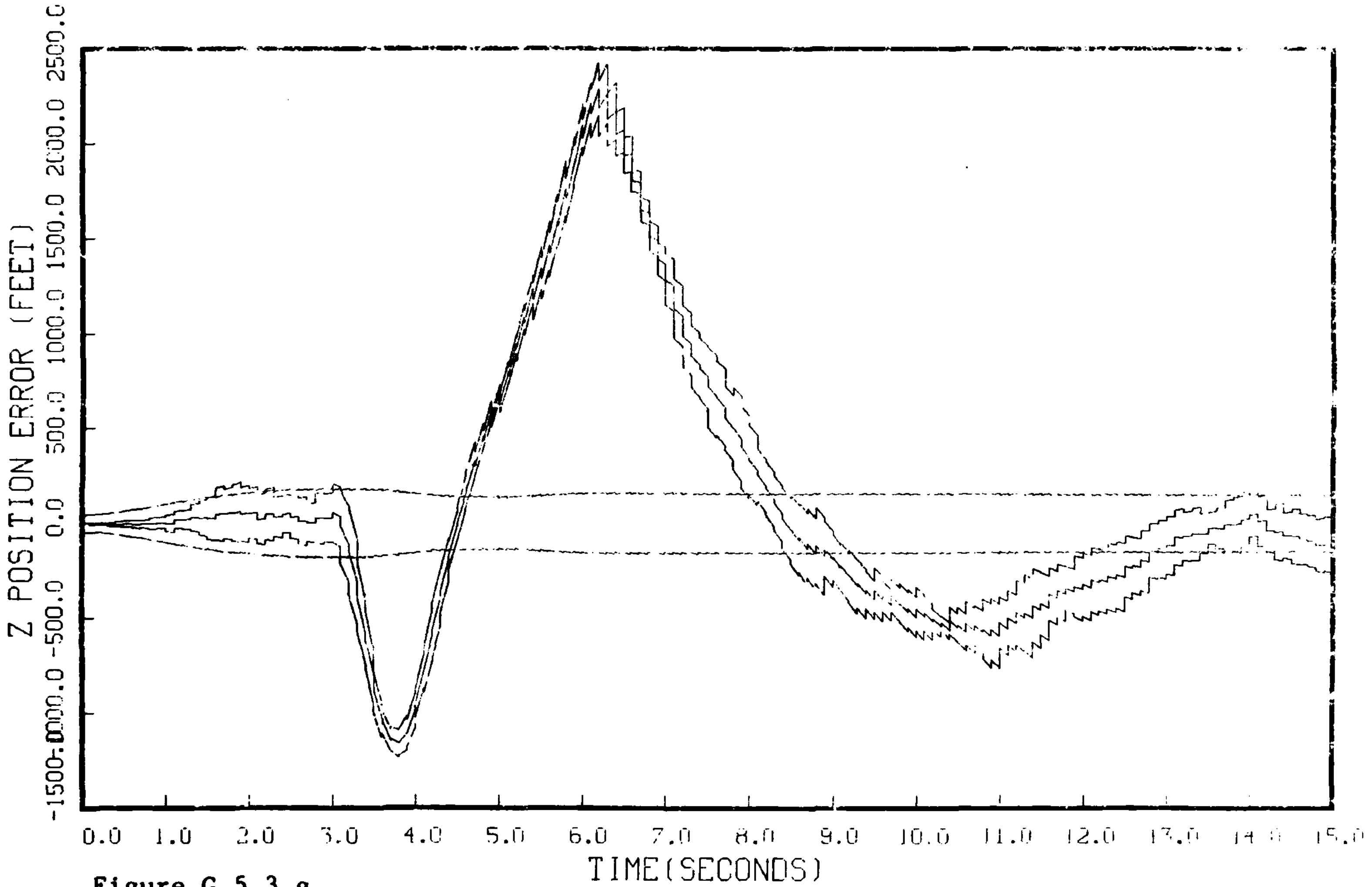


Figure G.5.3.g

STATE 7, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
ANG=120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

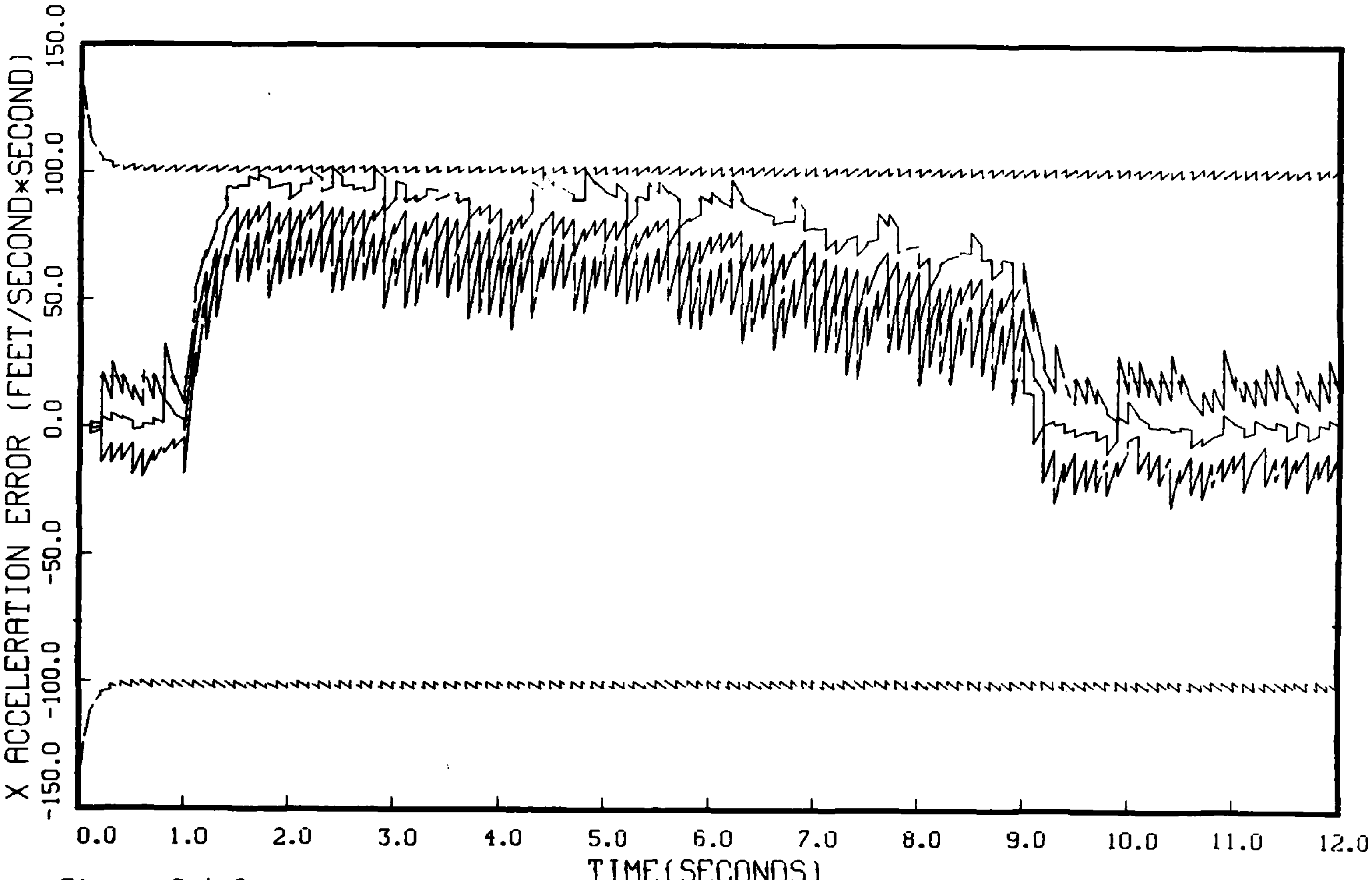


Figure G.4.5.c

STATE 3, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 20 RUNS

4 SOFEPL DATE AND TIME - 85/10/12. 14.35.26. SOFE DATE AND TIME - 85/10/12. 14.23.05. WPAFB, DISSPLA, SOFEPL VERSION 2.2

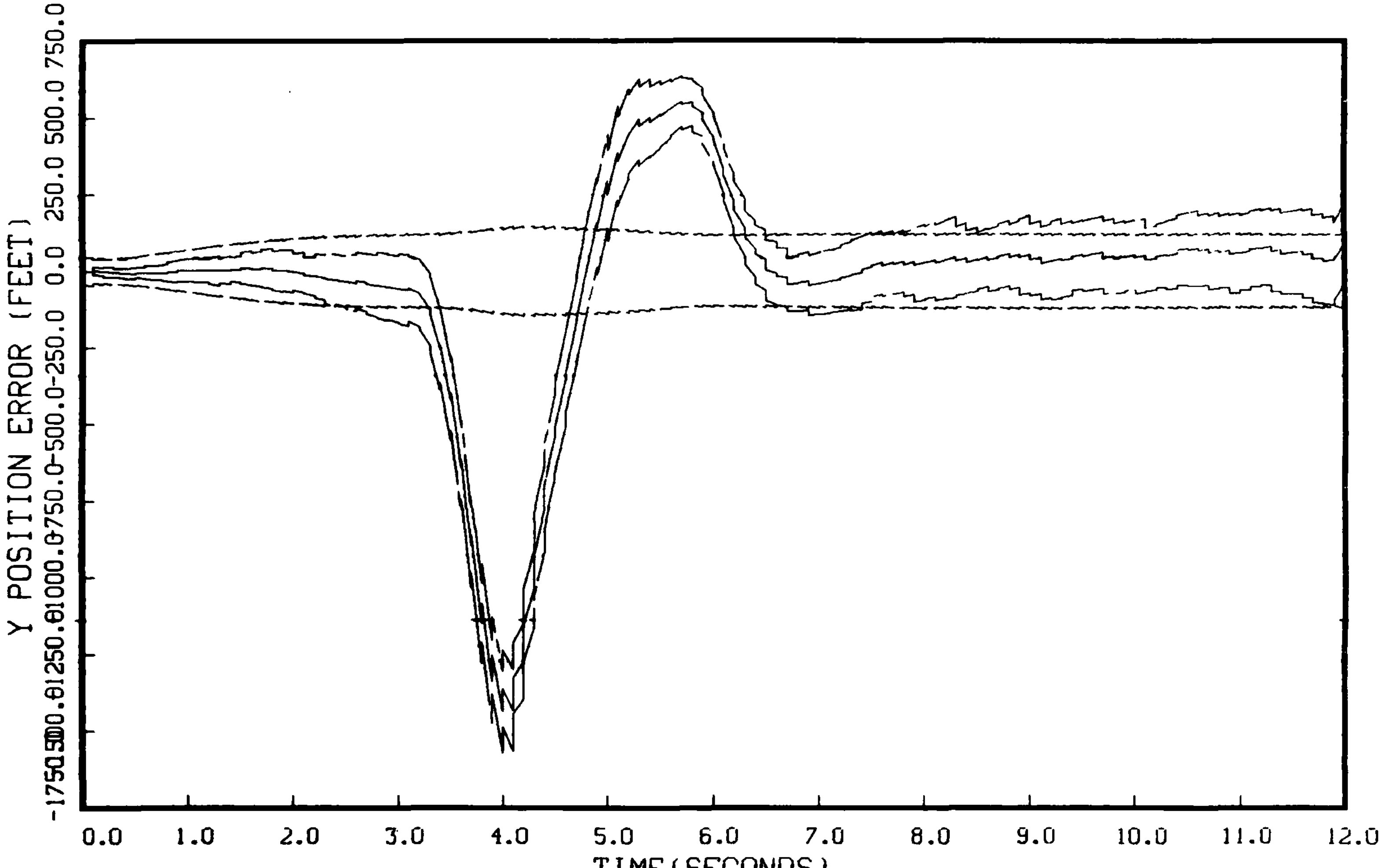


Figure G.4.5.d

STATE 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 20 RUNS

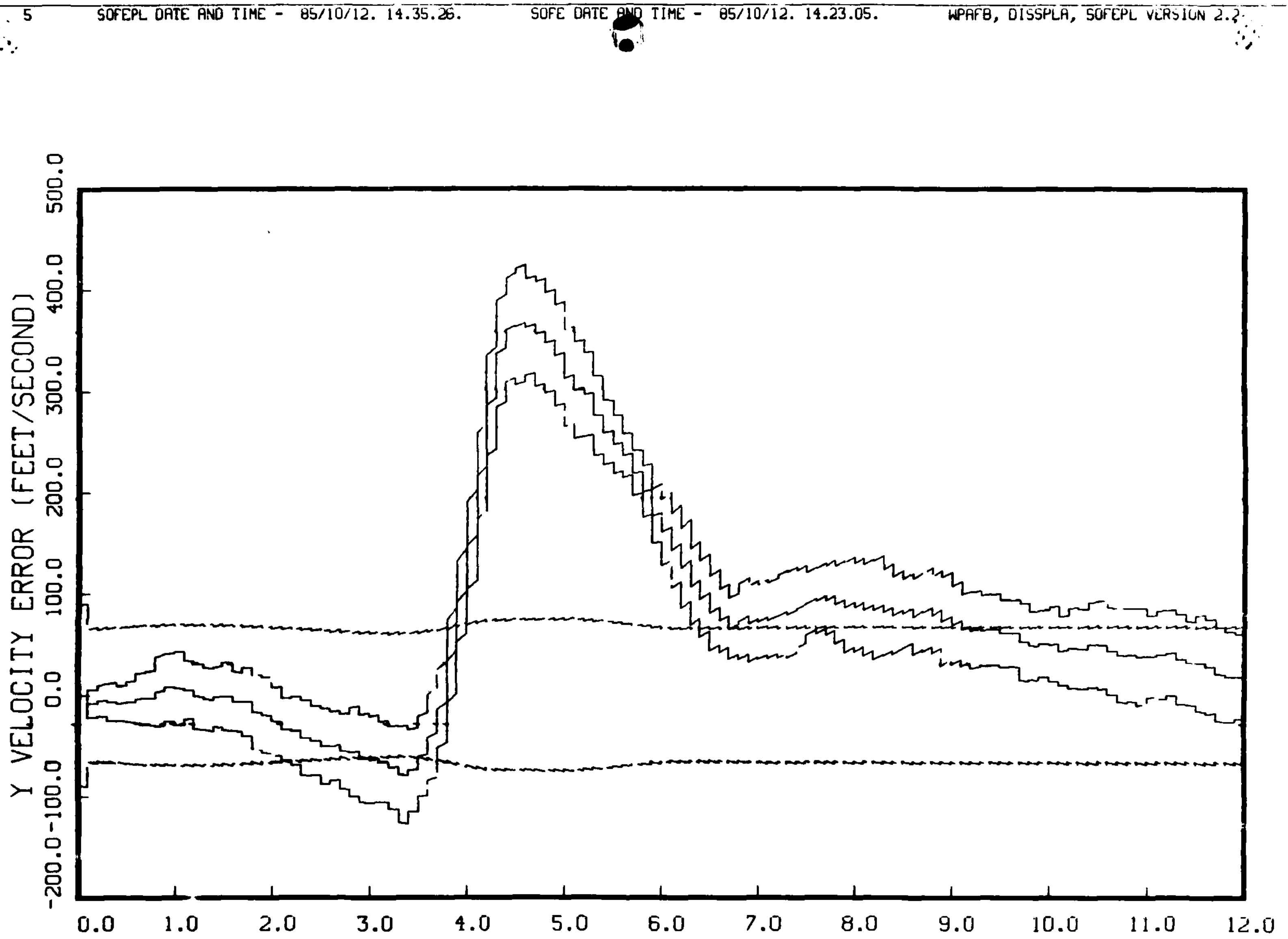


Figure G.4.5.e

STATE 5, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 20 RUNS

6 SOFEPL DATE AND TIME - 85/10/12. 14.35.26. SOFE DATE AND TIME - 85/10/12. 14.23.05. WPAFB, DISSPLA, SOFEPL VERSION 2.2

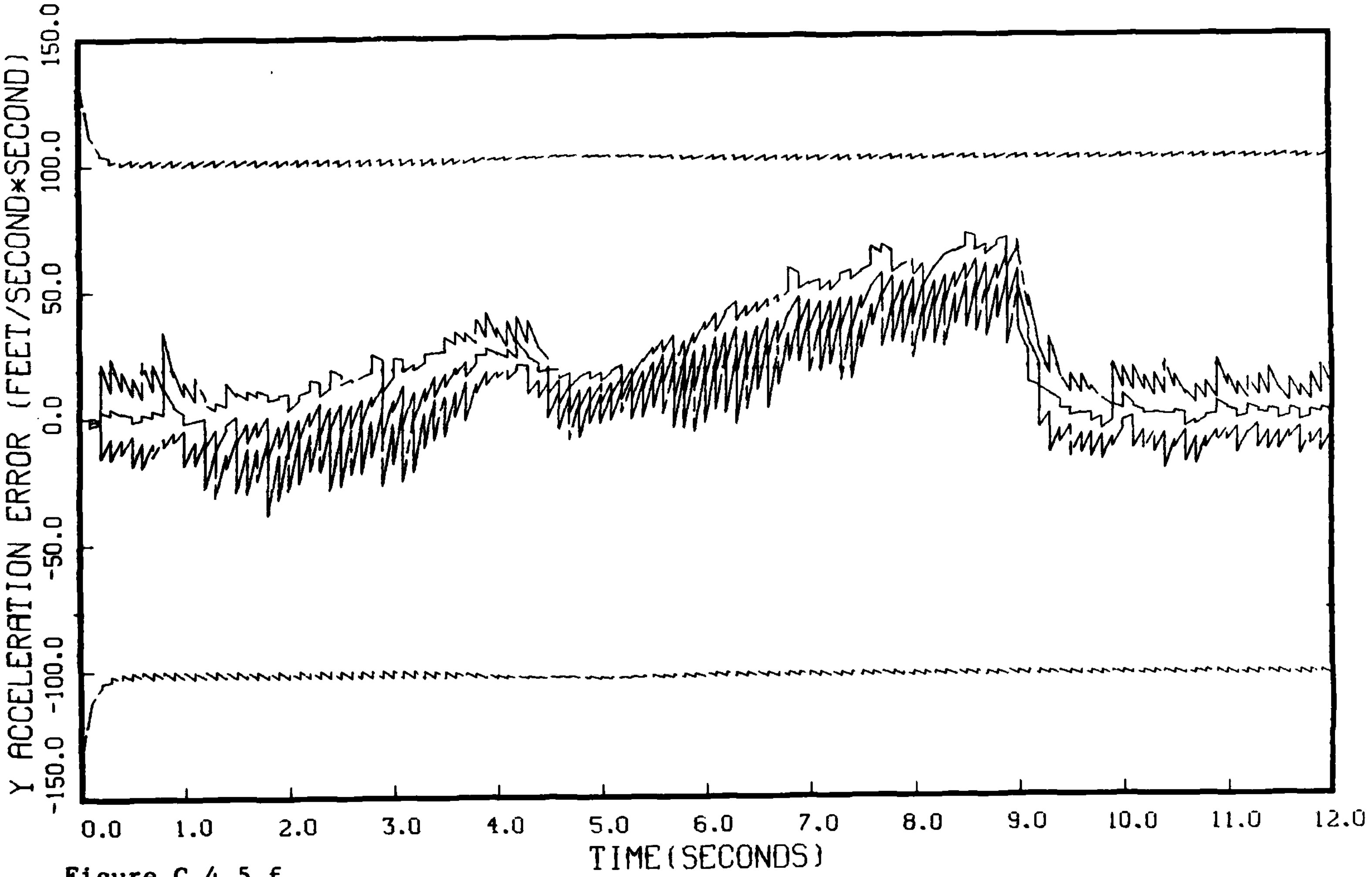


Figure G.4.5.f

STATE 6, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. ', UPDATE-0.1, 20 RUNS

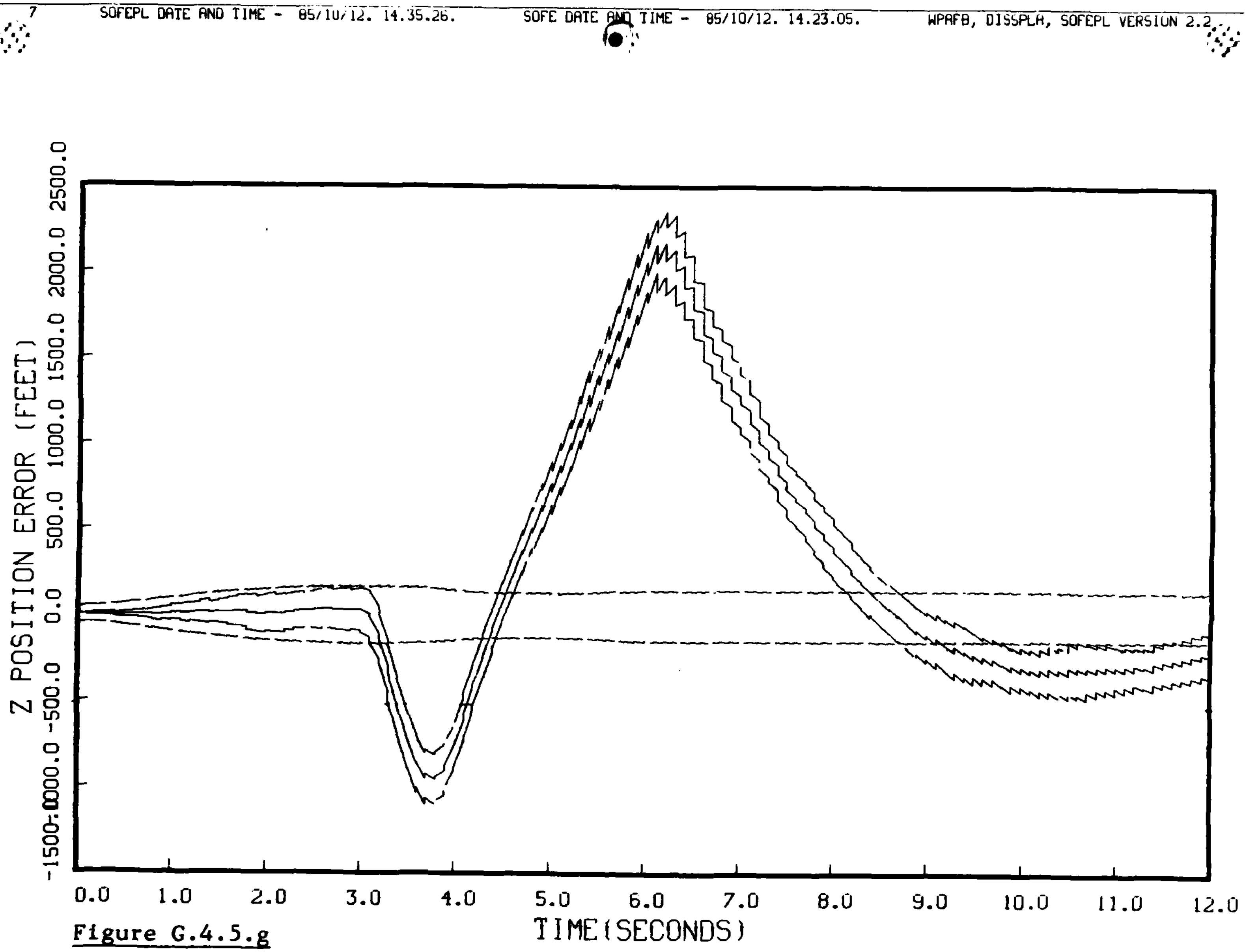


Figure G.4.5.g

STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 20 RUNS

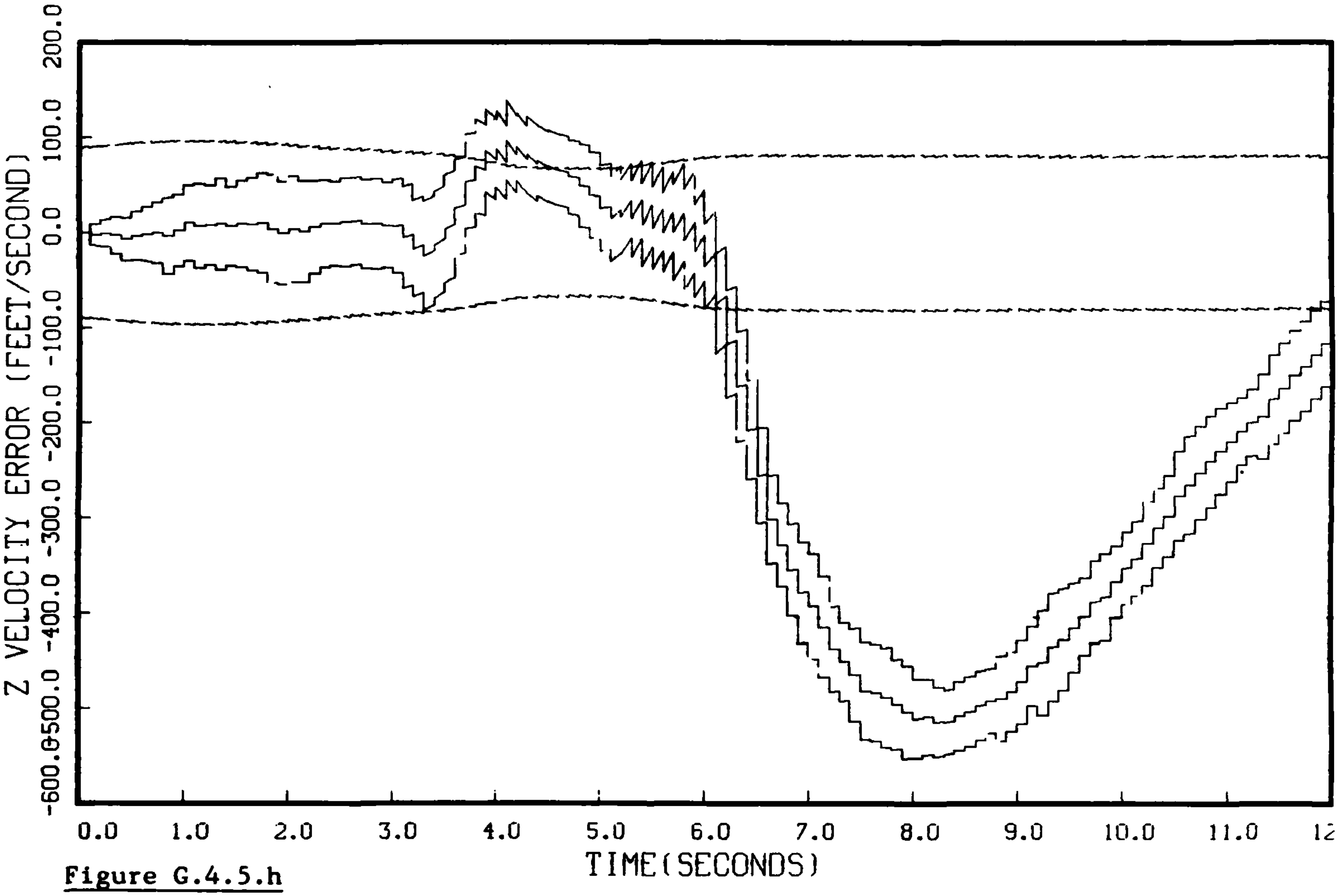


Figure G.4.5.h

STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 20 RUNS

9A PS4 #6 1 SOFEPL DATE AND TIME - 85/10/16. 22.09.31. SOFE DATE AND TIME - 85/10/16. 21.57.00. WPAFB, DISSPLA, SOFEPL VERSION 2.2

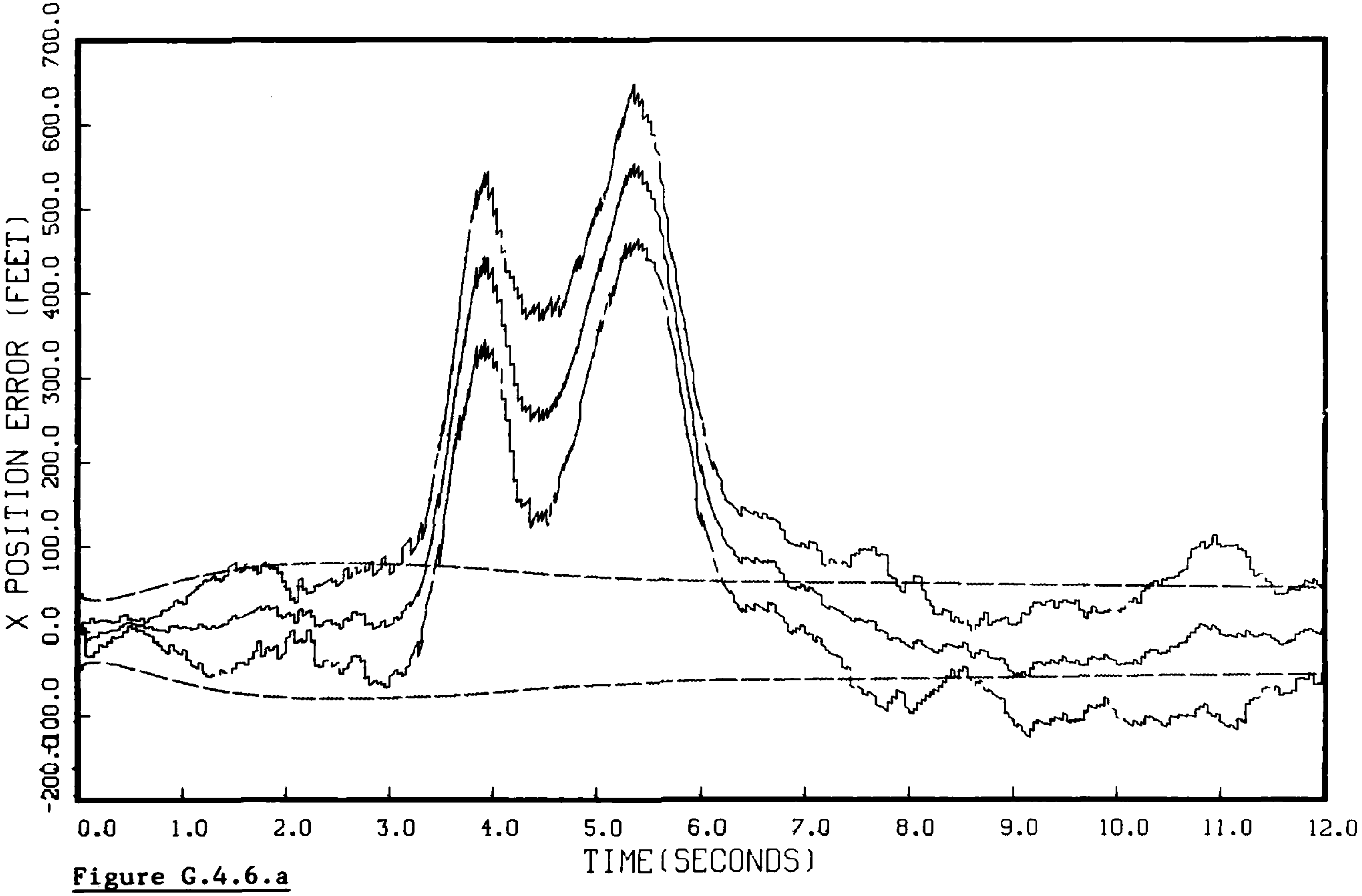


Figure G.4.6.a

STATE 1, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

2 SOFEPL DATE AND TIME - 85/10/16. 22.09.31. SOFE DATE AND TIME - 85/10/16. 21.57.00. WPAFB, DISSPLA, SOFEPL VERSION 2.2

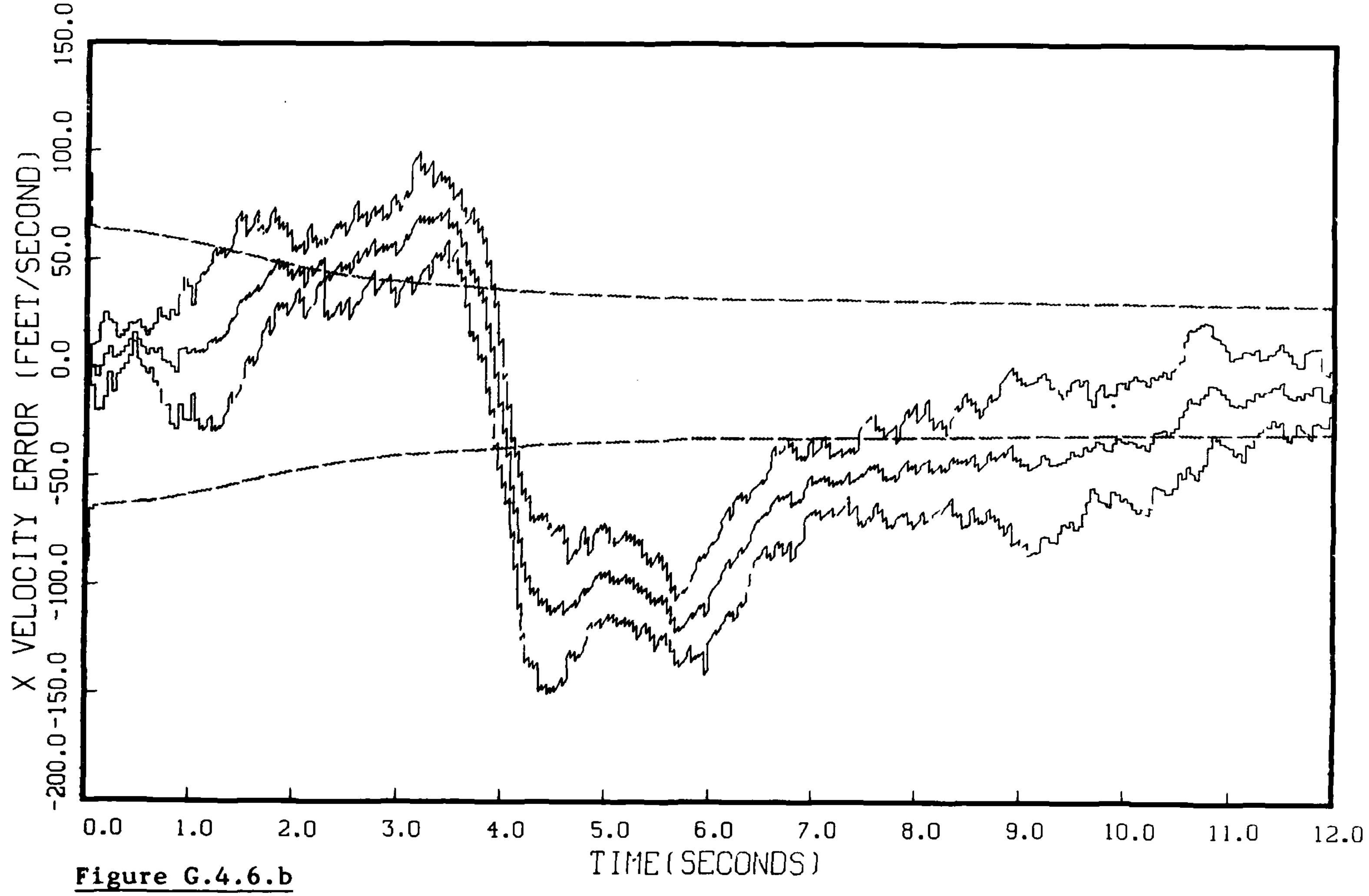


Figure G.4.6.b

STATE 2, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

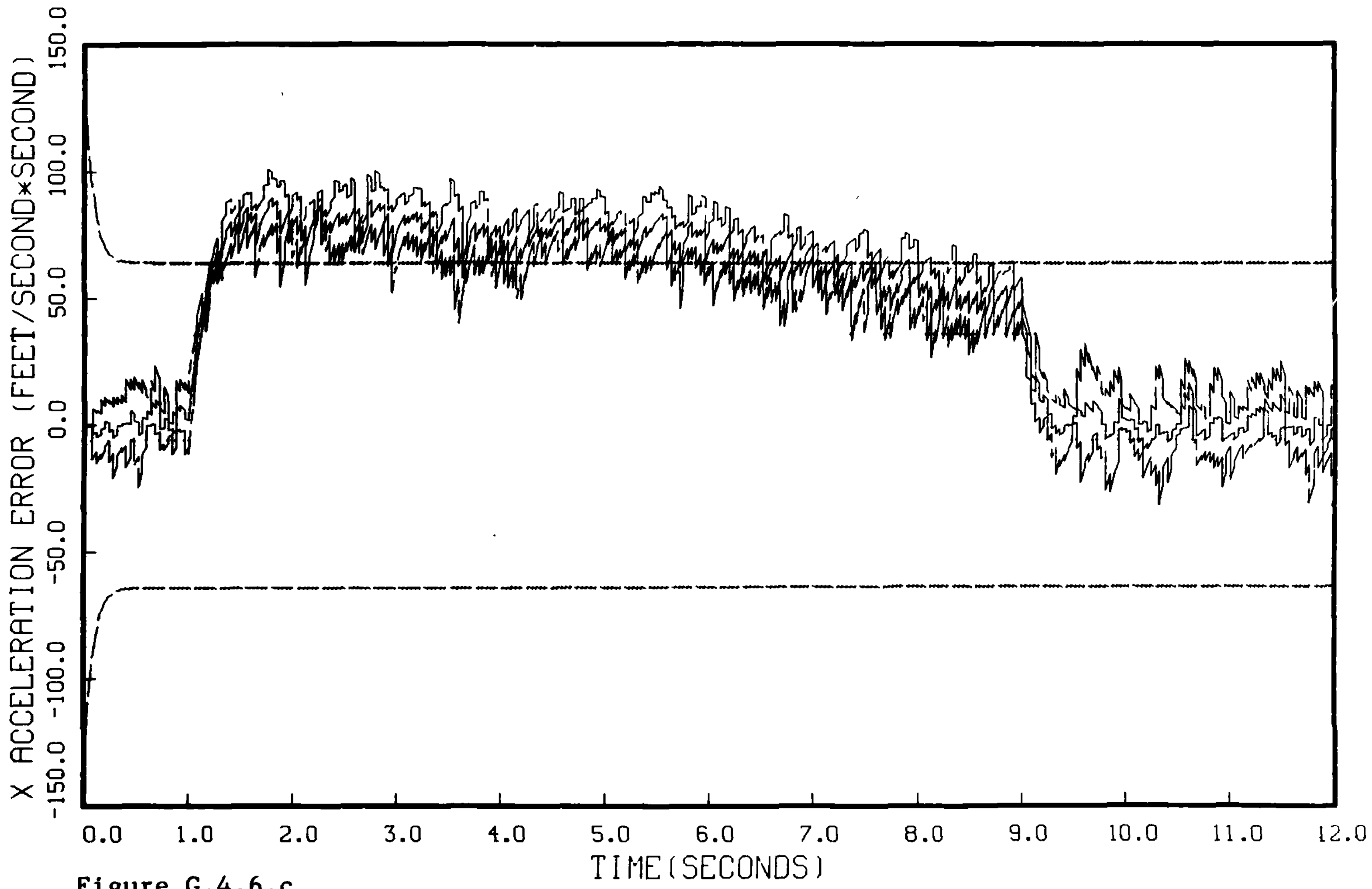


Figure G.4.6.c

STATE 3, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.04, 5 RUNS

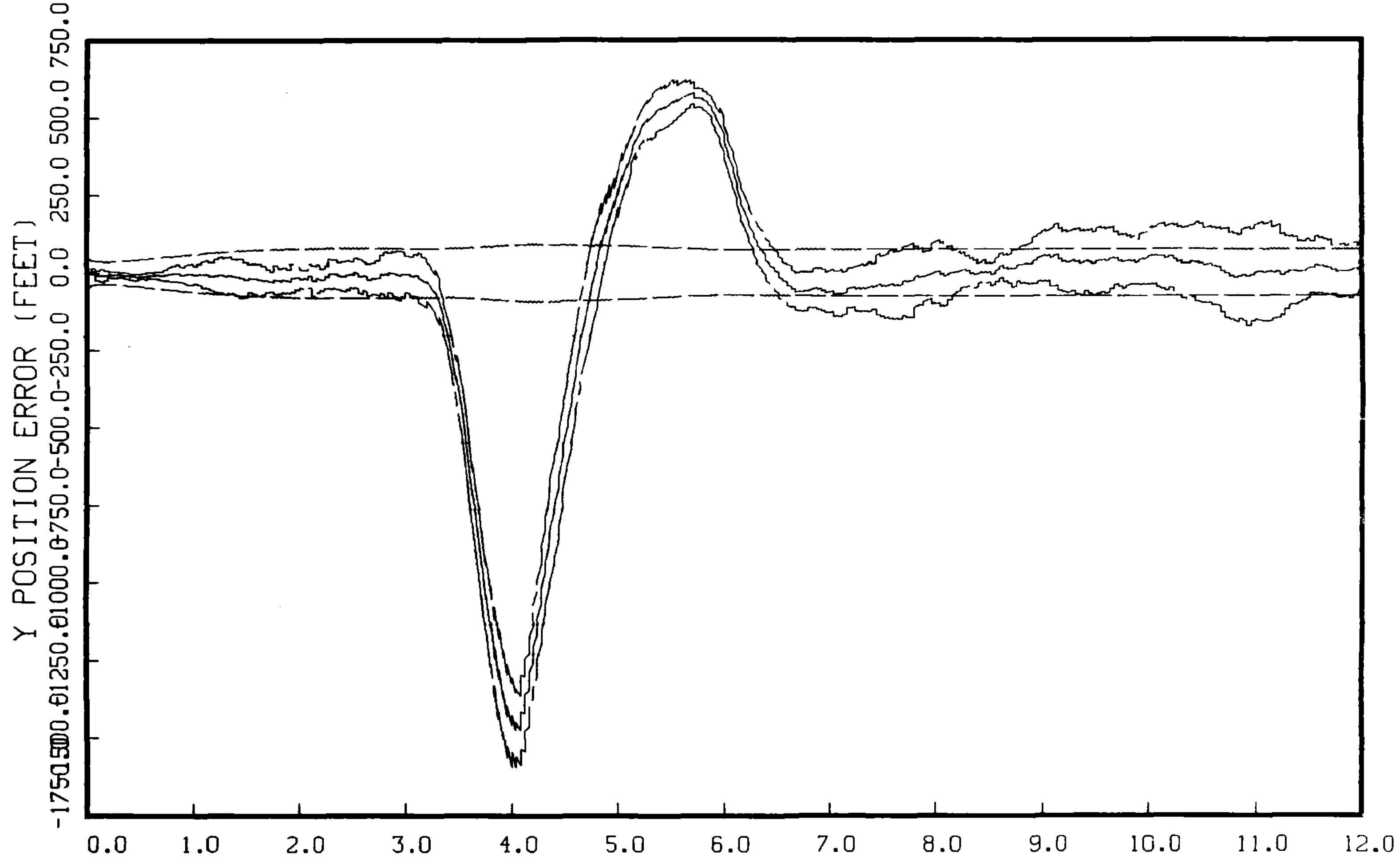


Figure G.4.6.d

STATE 4, Q(1)-Q(2)-Q(3)=59720., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

5 SOFEPL DATE AND TIME - 85/10/16. 22.09.31. SOFE DATE AND TIME - 85/10/16. 21.57.00. WPAFB, DISSPLA, SOFEPL VERSION 2.2

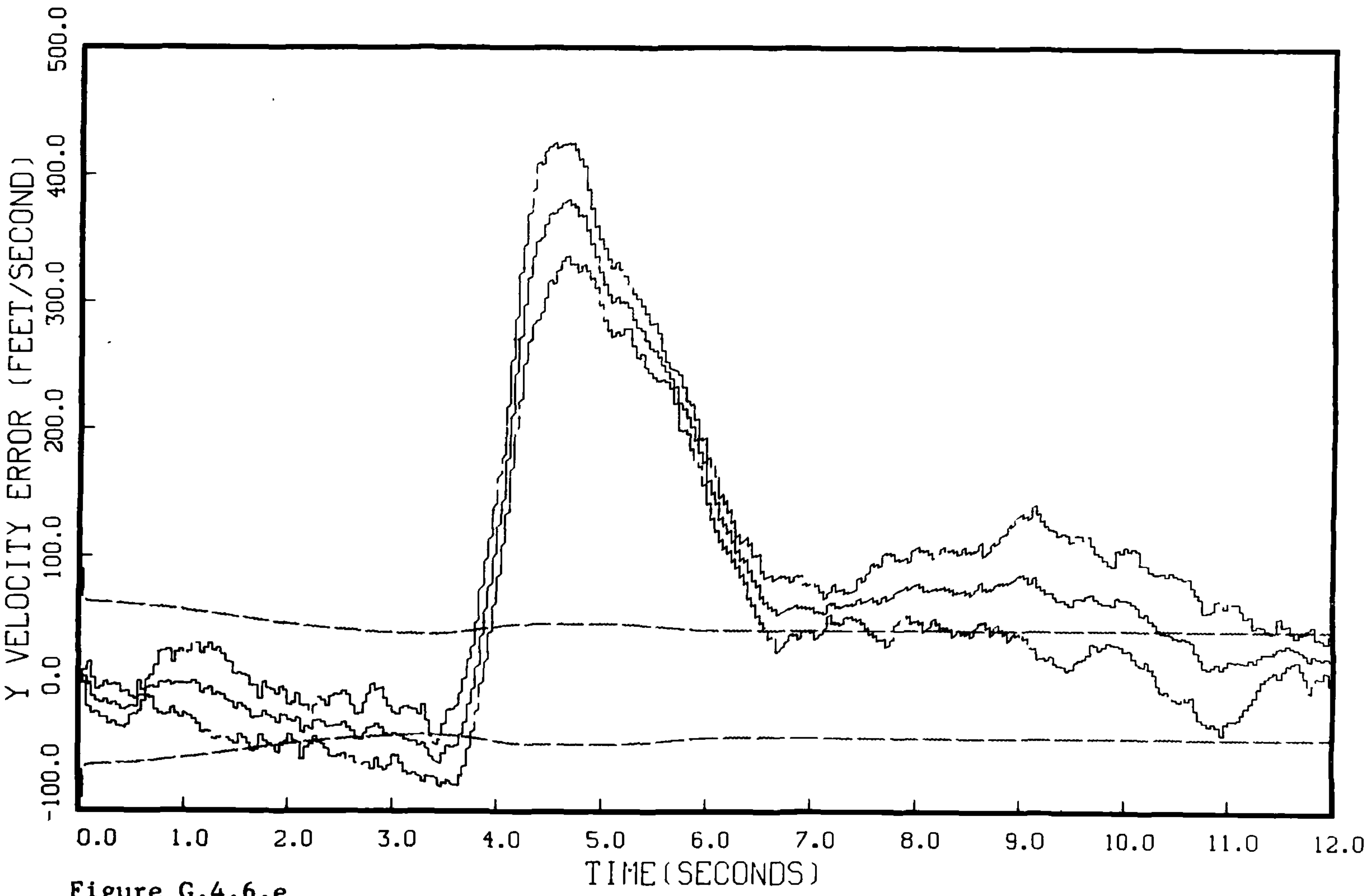


Figure G.4.6.e

STATE 5, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

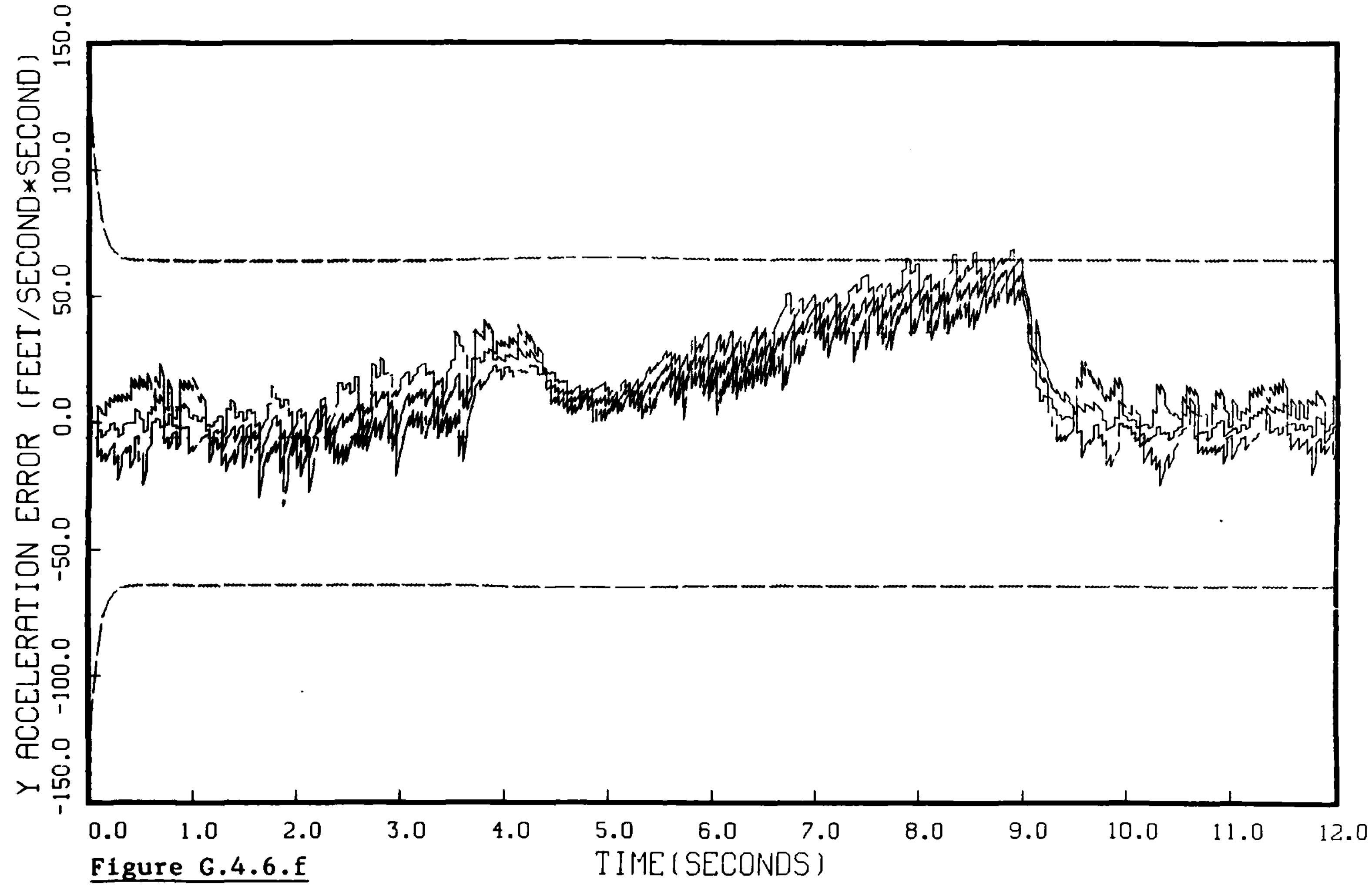


Figure G.4.6.f

STATE 6, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

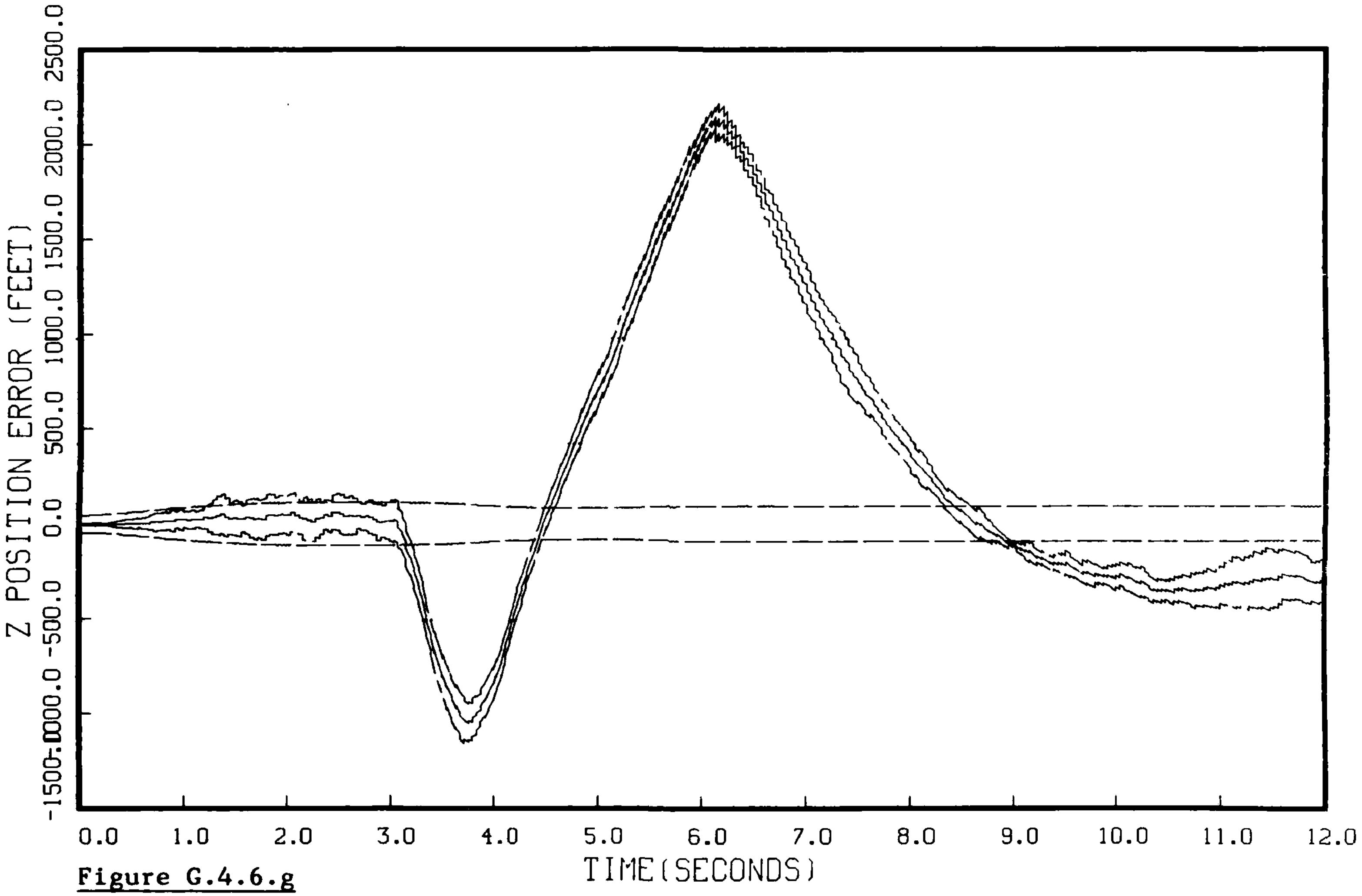


Figure G.4.6.g

STATE 7, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

8 SOFEPL DATE AND TIME - 85/10/16. 22.09.31. SOFE DATE AND TIME - 85/10/16. 21.57.00. WPAFB, DISSPLA, SOFEPL VERSION 2.2

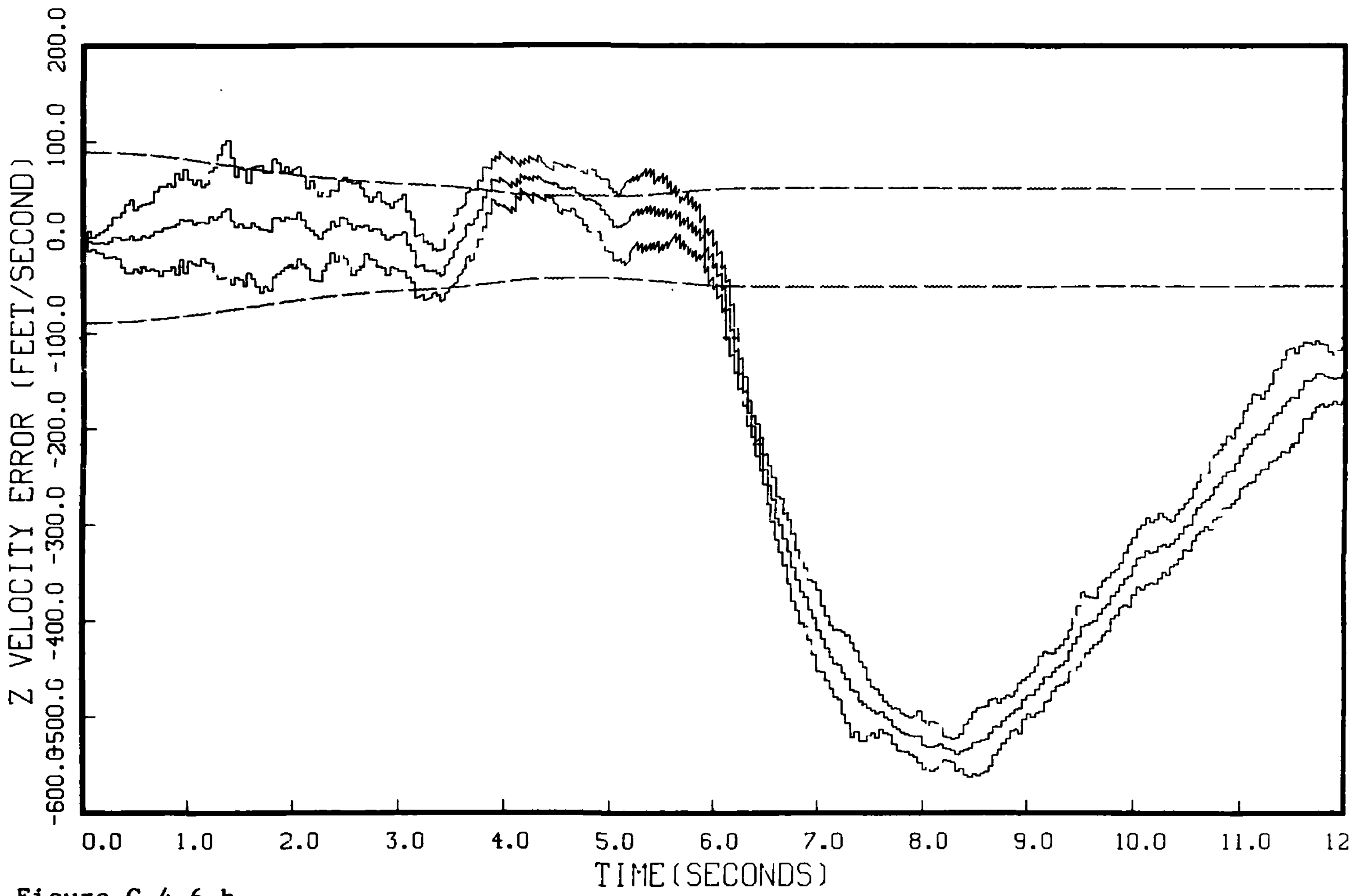


Figure G.4.6.h
STATE 8, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.04, 5 RUNS

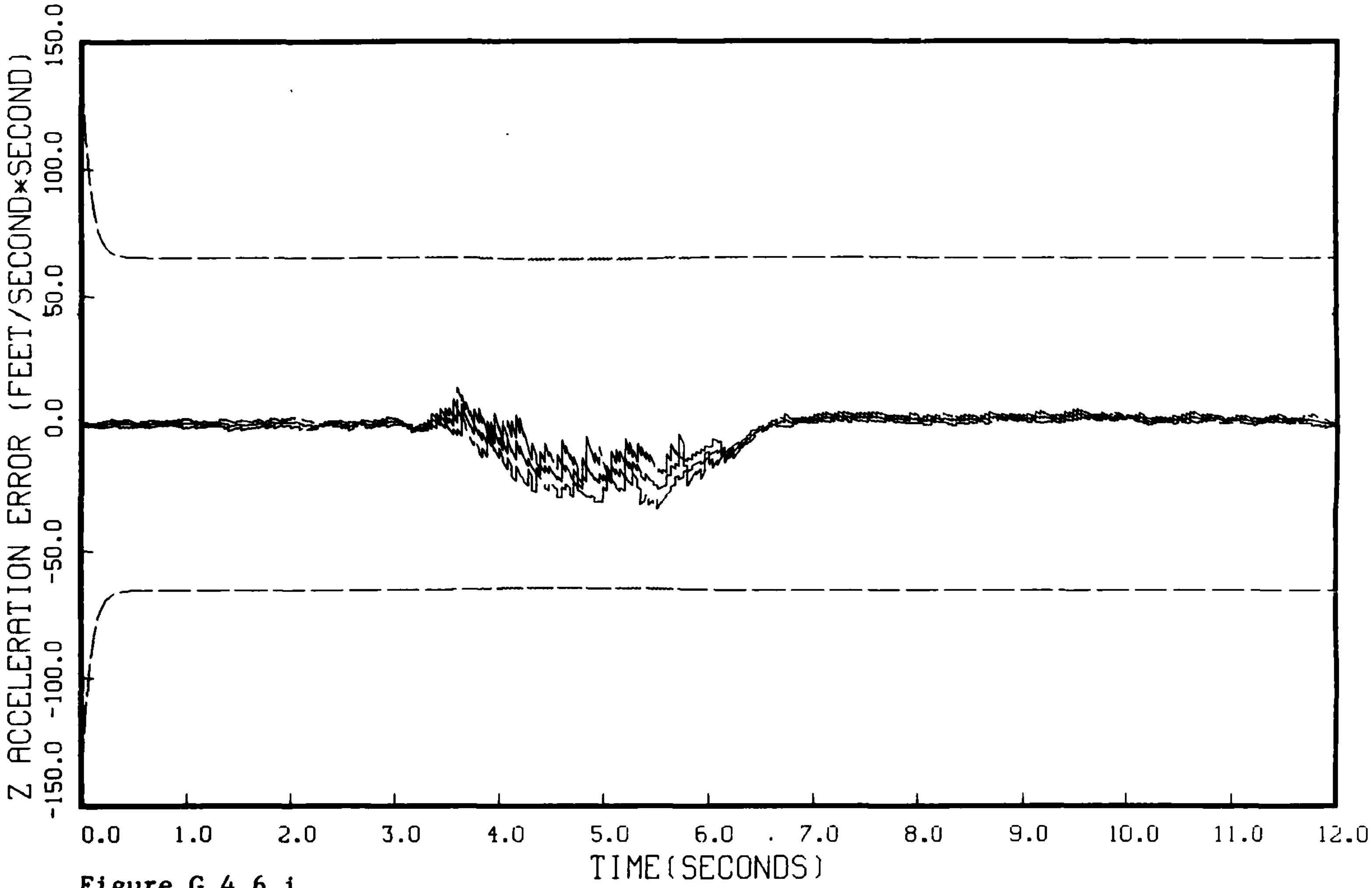


Figure G.4.6.i

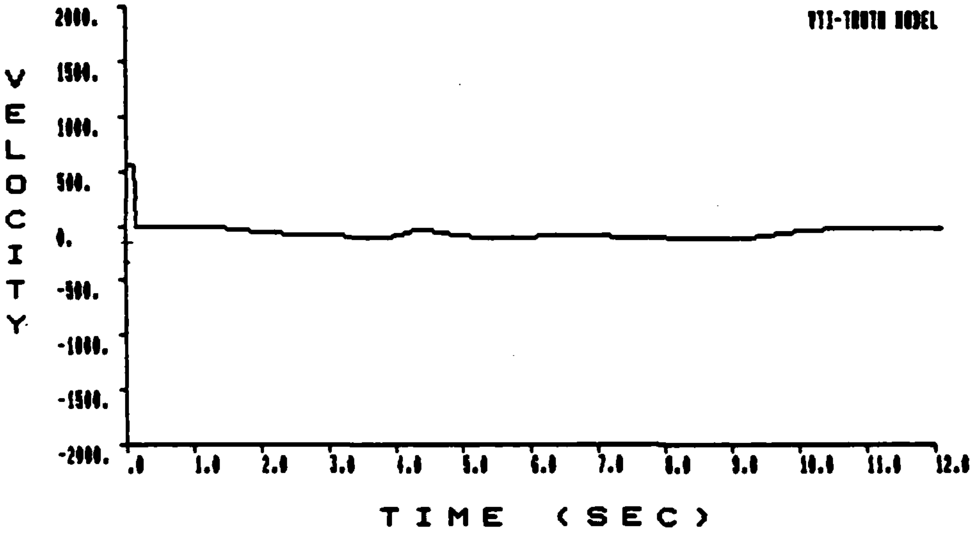
STATE 9, Q(1)-Q(2)-Q(3)-59720., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.04, 5 RUNS

Figure Set G.5.1 is the same as Figure Set G.3.2

```
***** APQ-120 RADAR MODEL--RADAR.FTN *****
      VERSION 25

TEST-RADAR MODEL TEST DATA SHORT RANGE-TRUTH MODEL
  MODE = ACM

  TIME STEP = 0.0400      ANGLE =  0.78540      RATE =  1.00000
  TRATE = -1.00000      ALPHA =      5.00      BETA =     10.00
  TR1 =    3.000   TF1 =    4.000   TR2 =    5.000   TF2 =    6.000
  TR3 =    0.000   TF3 =    0.000   TR4 =    0.000   TF4 =    0.000
  HORT1 =    1.000   HORT2 =    9.000   HORTG =  3.00   DELTA =      5.00
  RANGE =    40000.   ASPECT =  0.78540   TAS =    800.0   TAST =    800.0
  SN1 = 0.0000    SN2 = 0.0000    SN3 = 0.0000    FR = 18.85
```



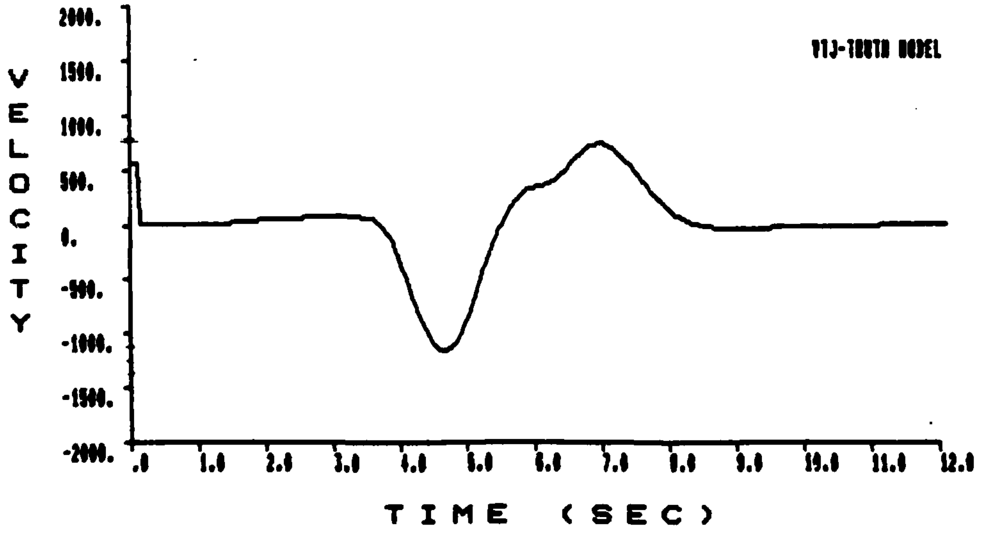


Figure G.5.2.a

```
***** APQ-120 RADAR MODEL--RADAR.FTN *****
      VERSION 25

TEST-RADAR MODEL TEST DATA SHORT RANGE-TRUTH MODEL
  MODE = ACM

  TIME STEP = 0.0400      ANGLE =  0.78540      RATE =  1.00000
  TRATE = -1.00000      ALPHA =     5.00      BETA =     10.00
  TR1 =    3.000    TF1 =    4.000    TR2 =    5.000    TF2 =    6.000
  TR3 =    0.000    TF3 =    0.000    TR4 =    0.000    TF4 =    0.000
  HORT1 =    1.000    HORT2 =    9.000    HORTG =  3.00    DELTA =      5.00
  RANGE =    40000.    ASPECT =  0.78540    TAS =    800.0    TAST =    800.0
  SN1 = 0.0000    SN2 = 0.0000    SN3 = 0.0000    FR = 18.85
```

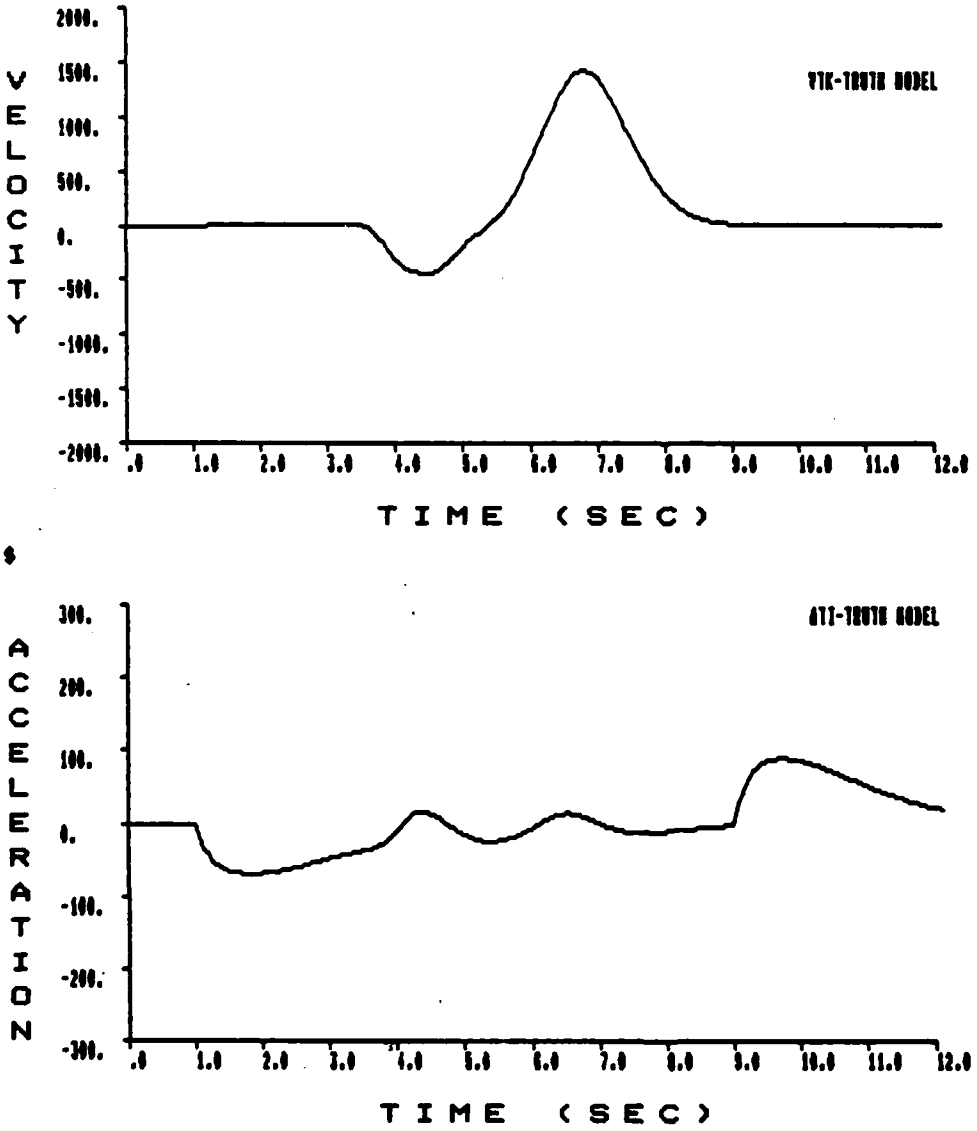
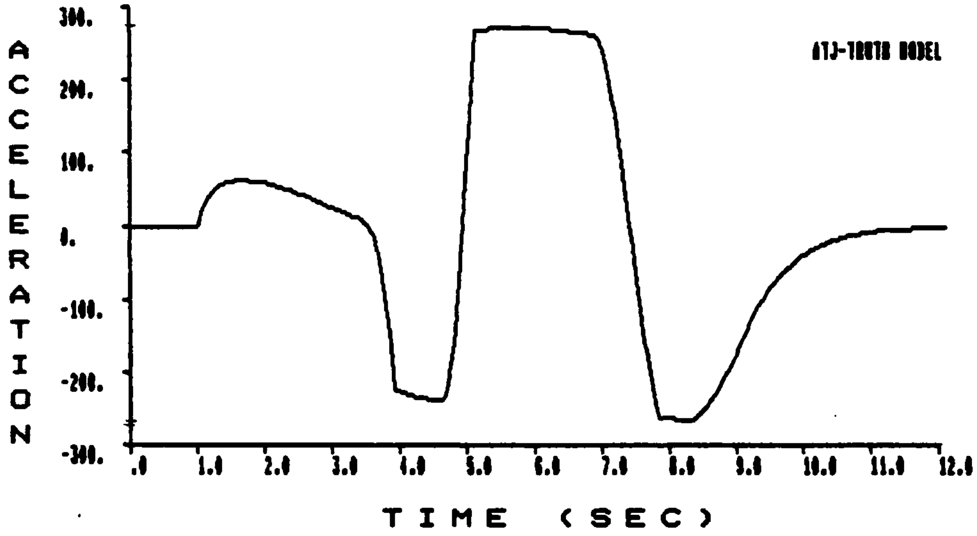


Figure G.5.2.b

```
***** APQ-120 RADAR MODEL--RADAR.FTN *****
      VERSION 25

TEST-RADAR MODEL TEST DATA SHORT RANGE-TRUTH MODEL
  MODE = ACM

  TIME STEP = 0.0400      ANGLE =  0.78540     RATE =  1.00000
  TRATE = -1.00000     ALPHA =     5.00     BETA =    10.00
  TR1 =   3.000   TF1 =   4.000   TR2 =   5.000   TF2 =   6.000
  TR3 =   0.000   TF3 =   0.000   TR4 =   0.000   TF4 =   0.000
  HORT1 =   1.000   HORT2 =   9.000   HORTG =  3.00   DELTA =      5.00
  RANGE =   40000.   ASPECT =  0.78540   TAS =   800.0   TAST =   800.0
  SN1 = 0.0000   SN2 = 0.0000   SN3 = 0.0000   FR = 18.85
```



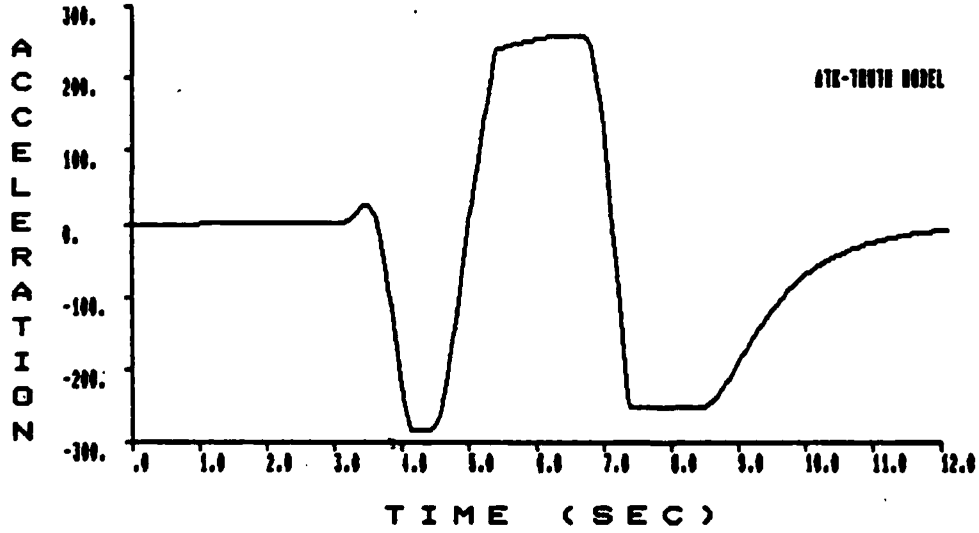


Figure G.5.2.c

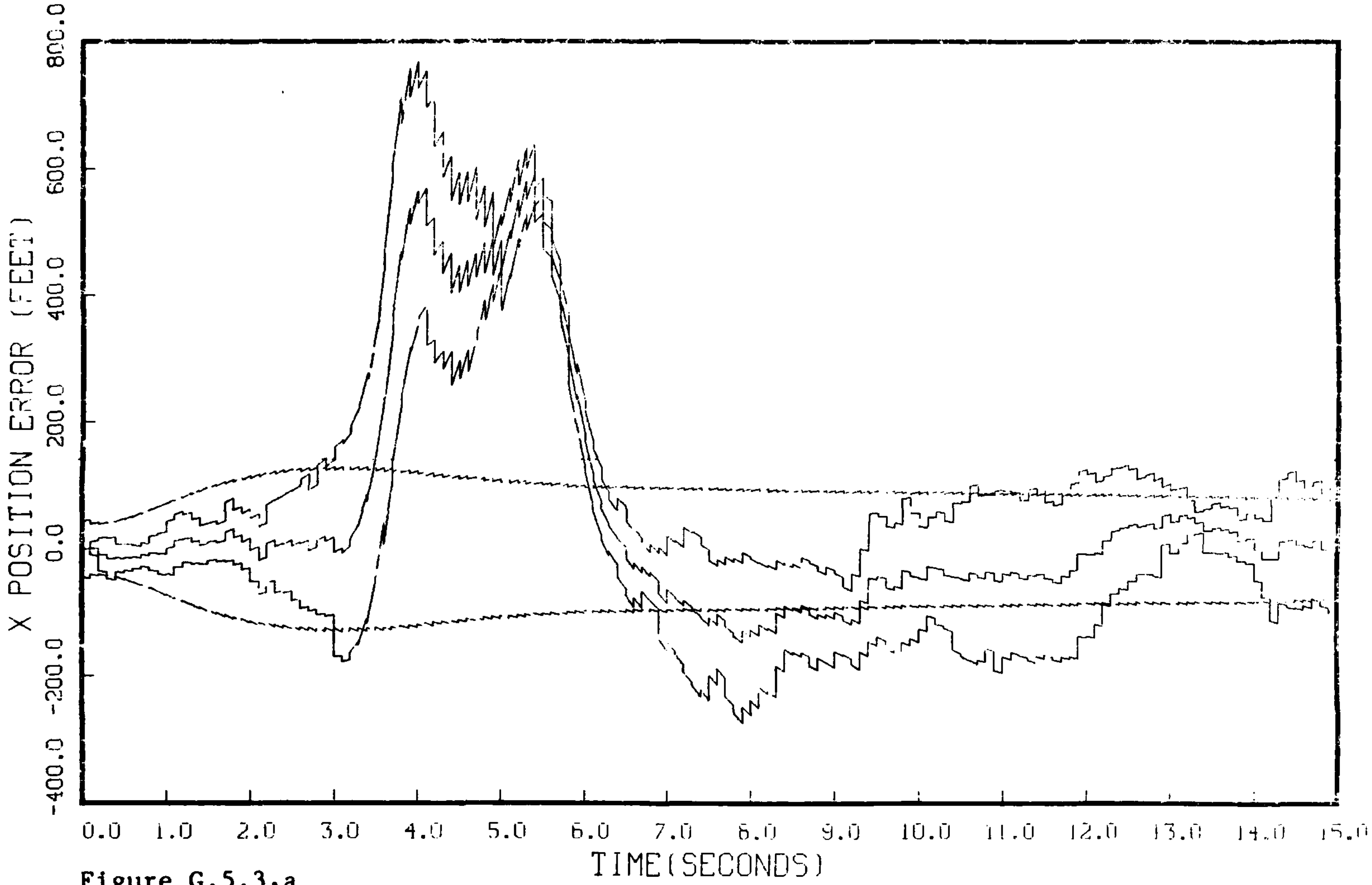


Figure G.5.3.a

STATE 1, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

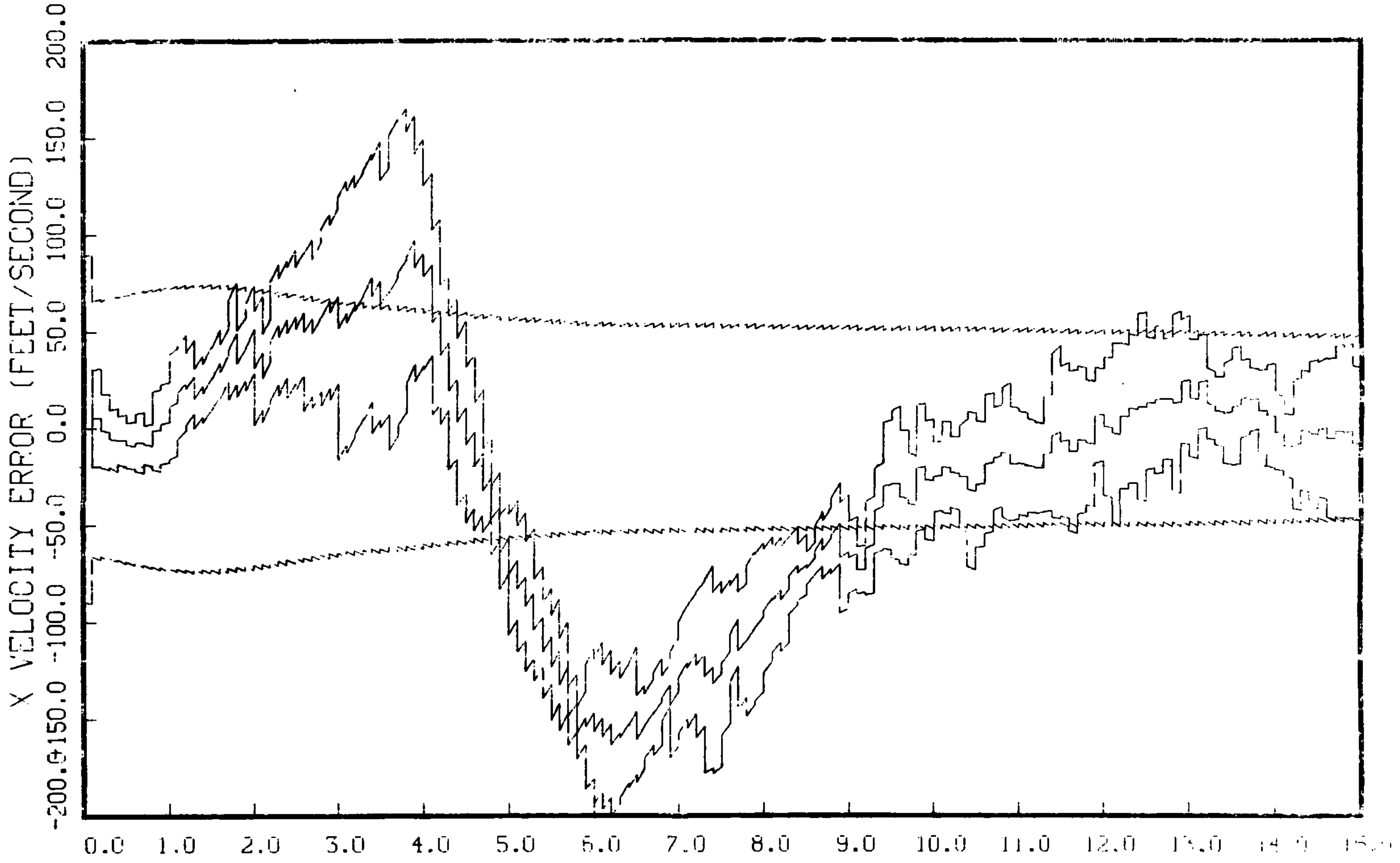


Figure G.5.3.b

STATE 2, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

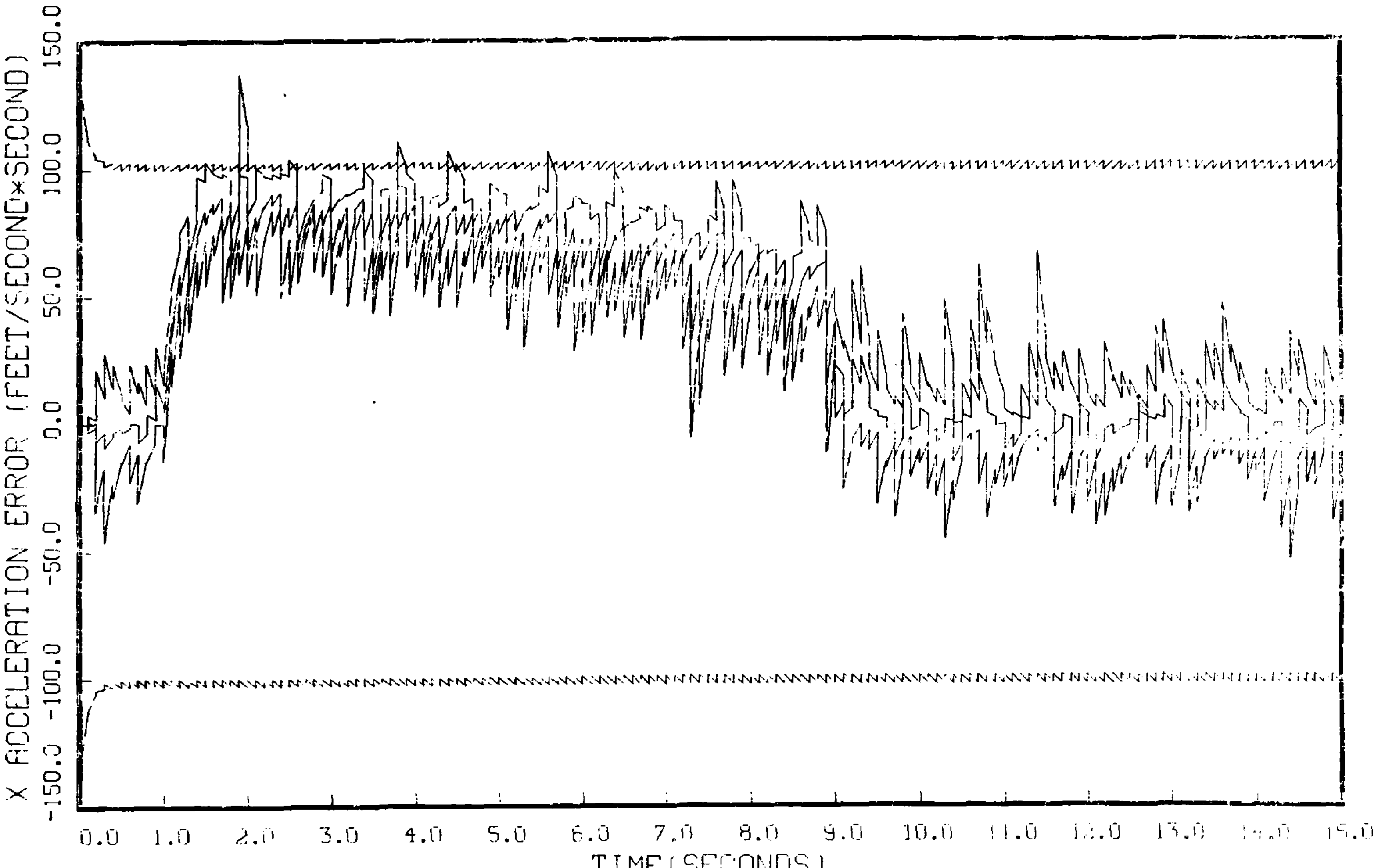


Figure G.5.3.c

STATE 3, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APG 120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

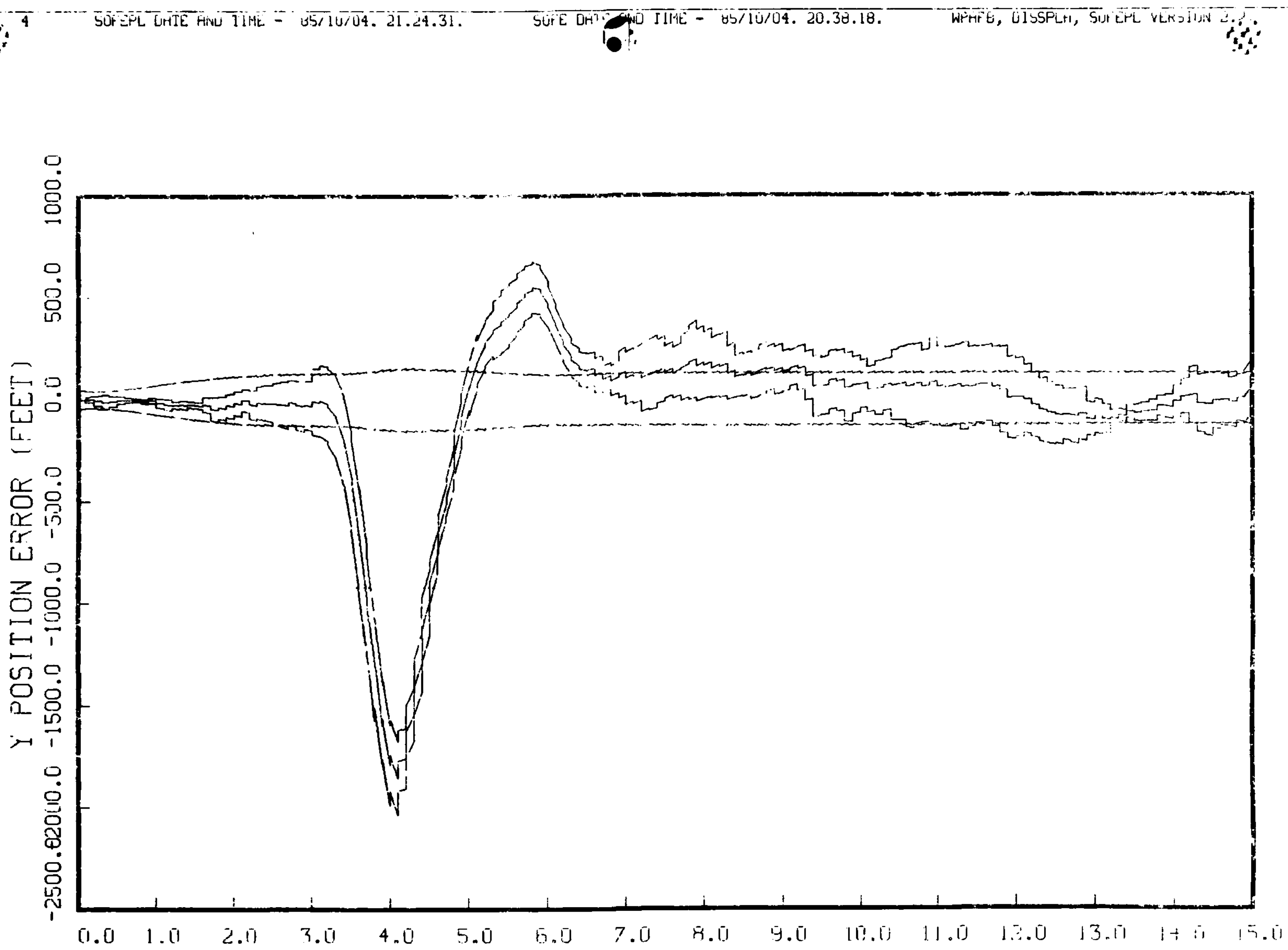


Figure G.5.3.d

STATE 4, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,ROOT,AZ,EL MEAS
APG 120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

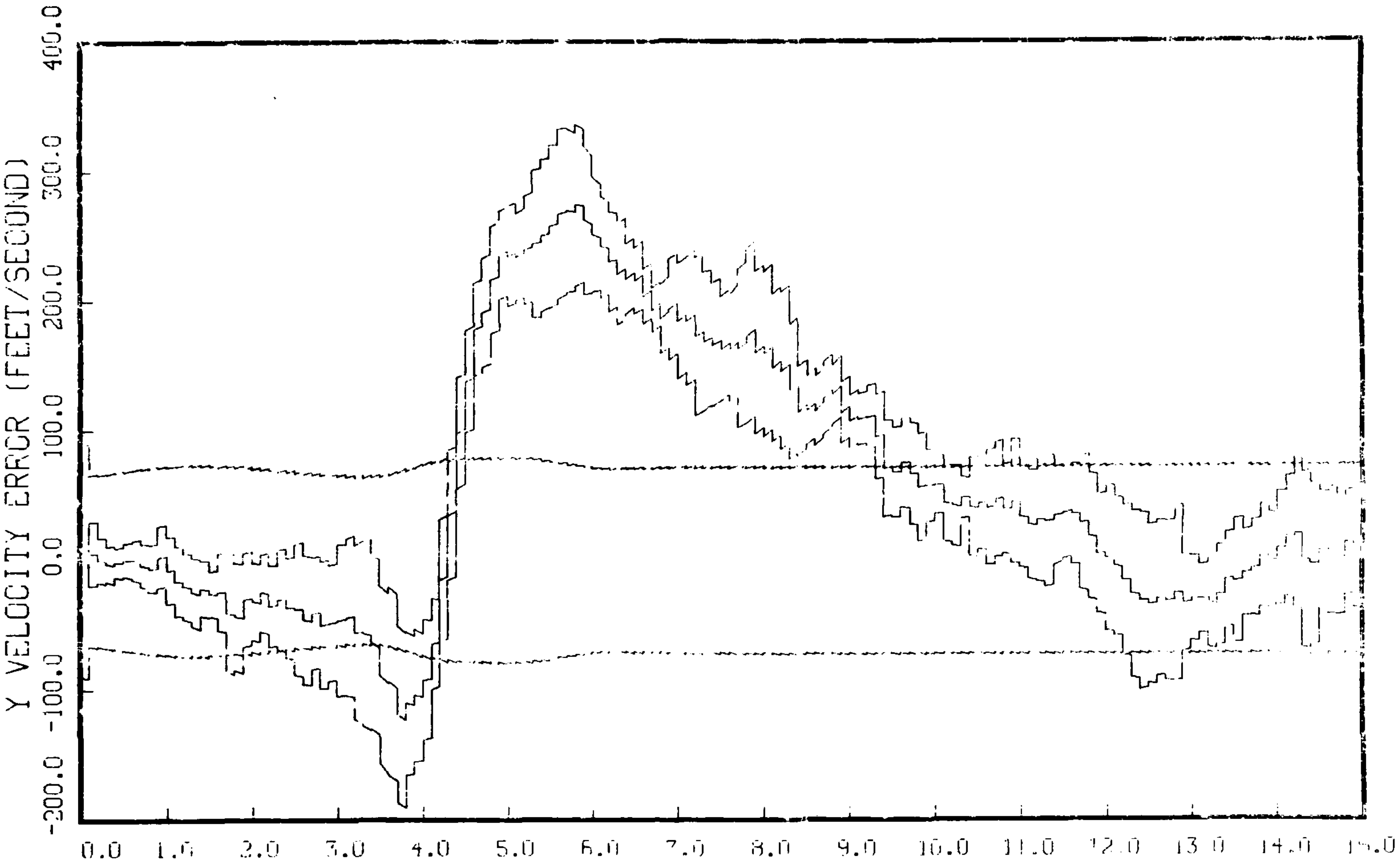


Figure G.5.3.e
STATE 5, Q(1)-Q(2)-Q(3)-149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APU 120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS

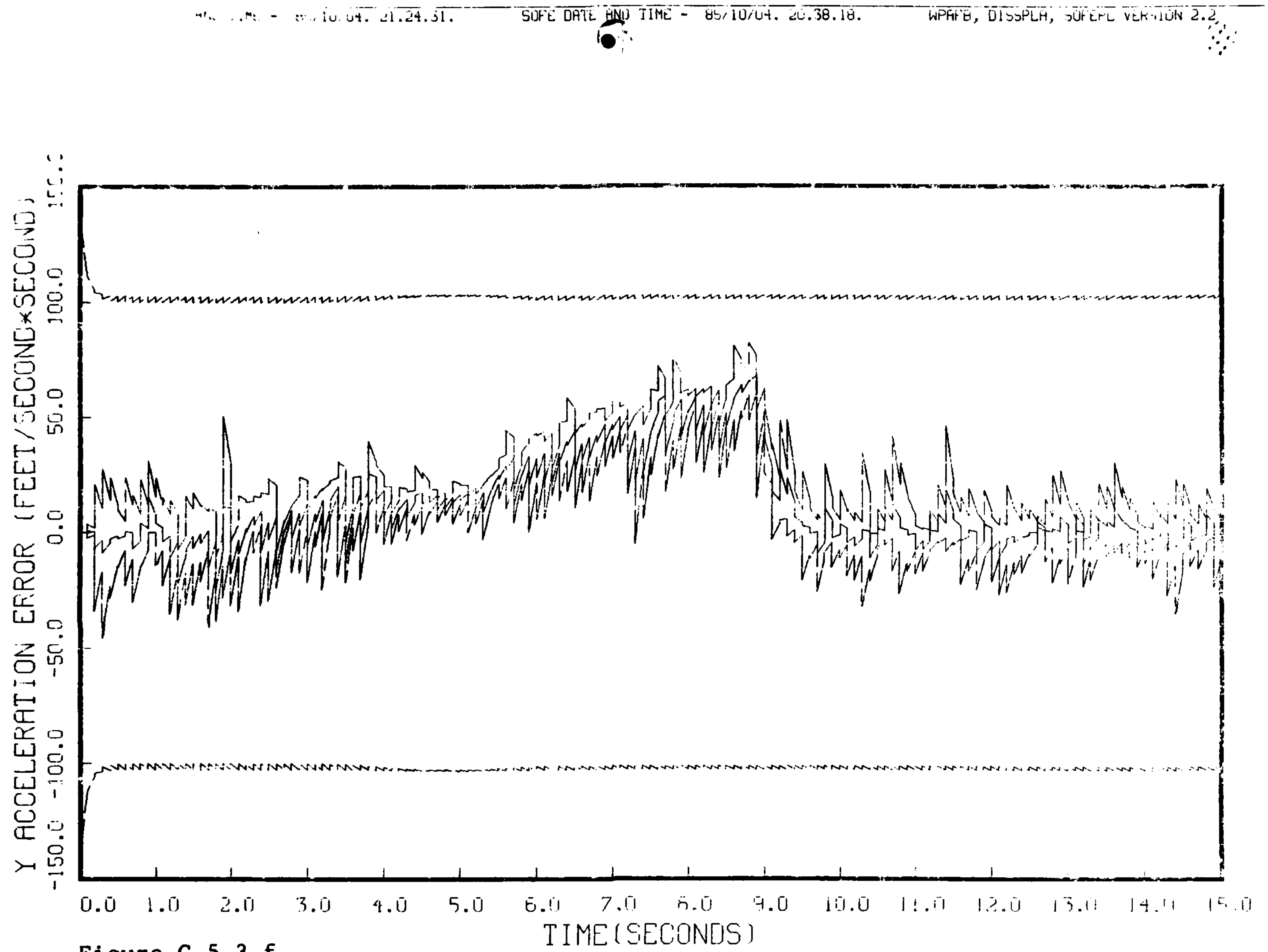


Figure G.5.3.f

STATE 6, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

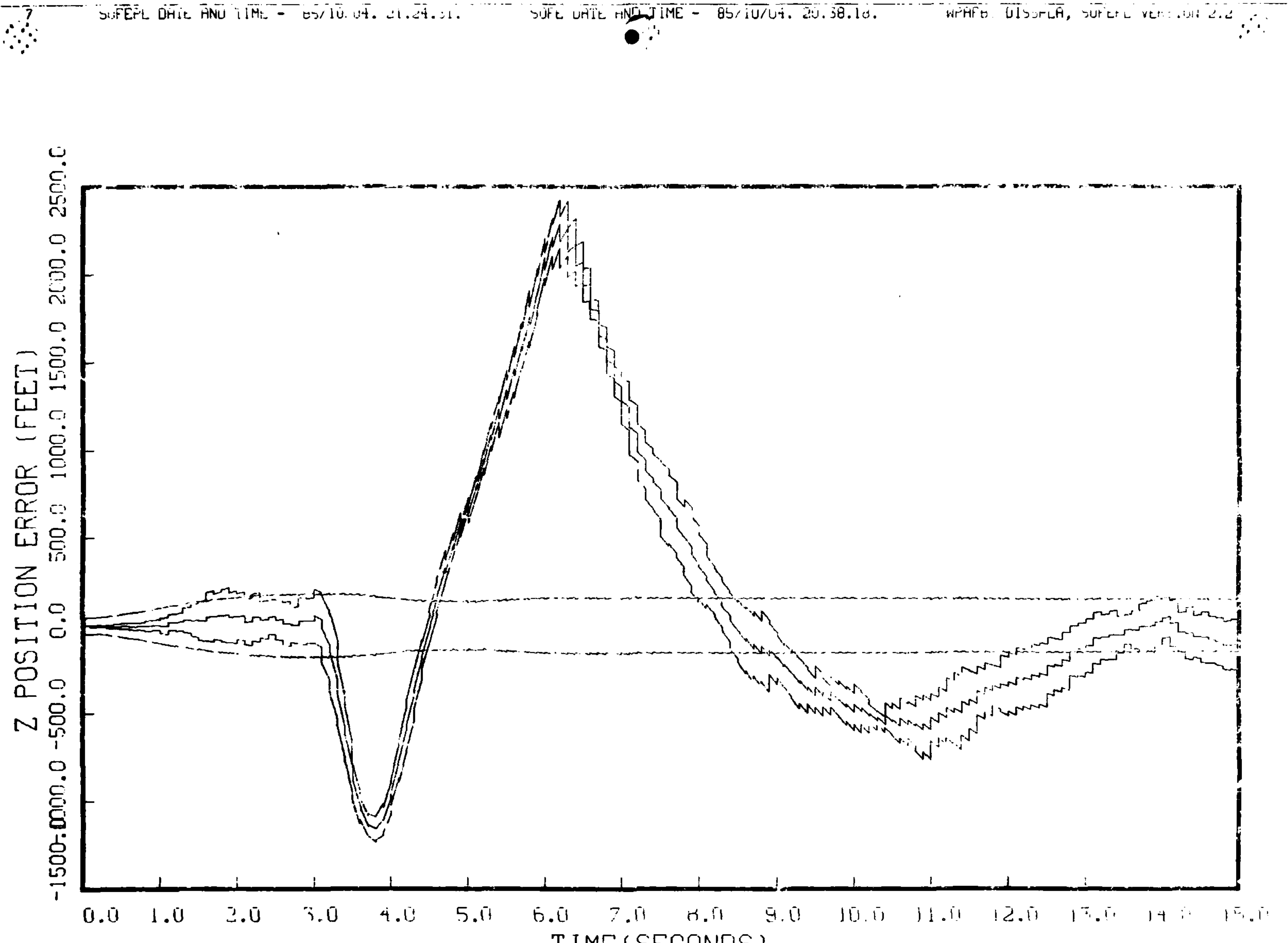


Figure G.5.3.g

STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

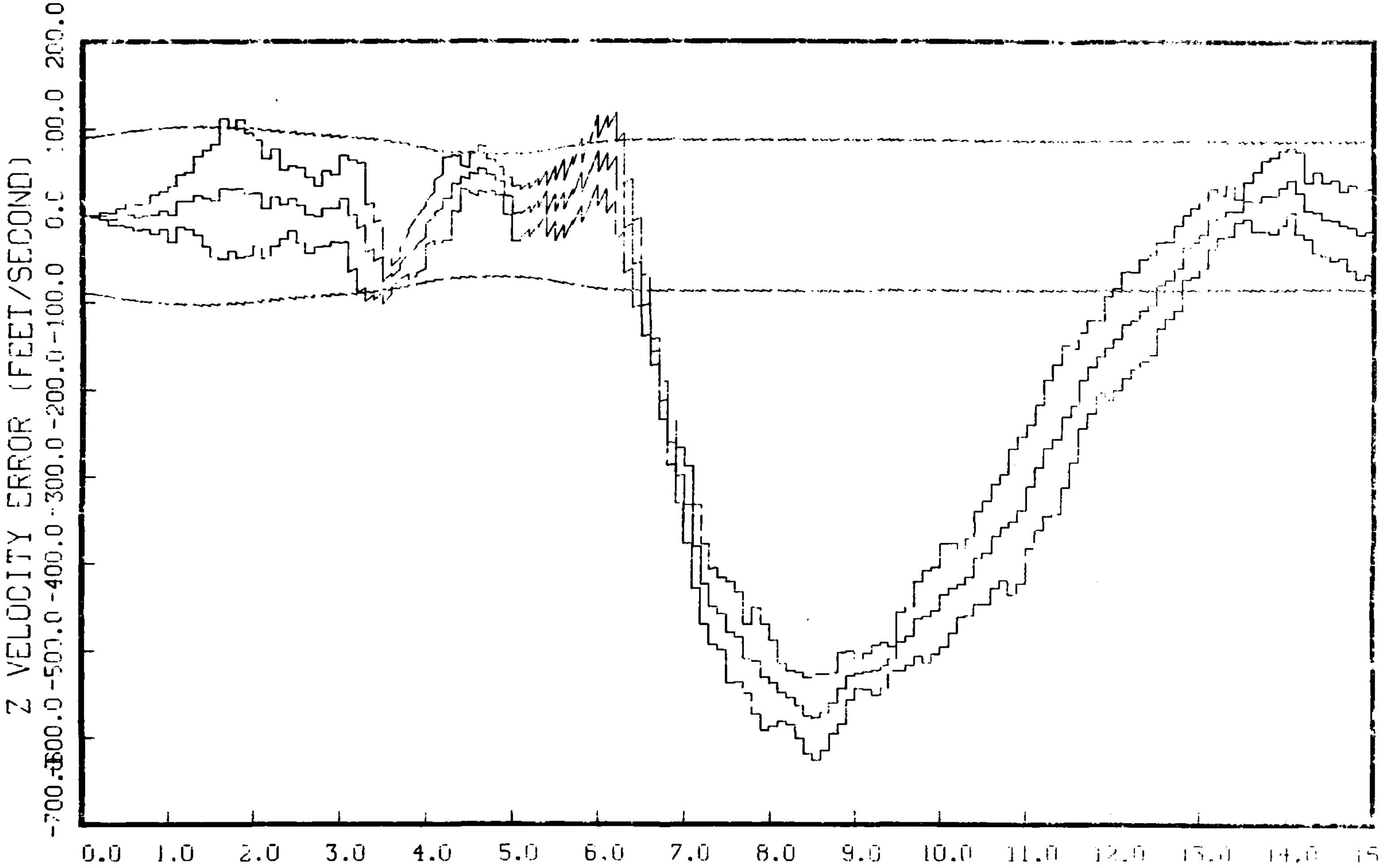


Figure G.5.3.h

STATE 8, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

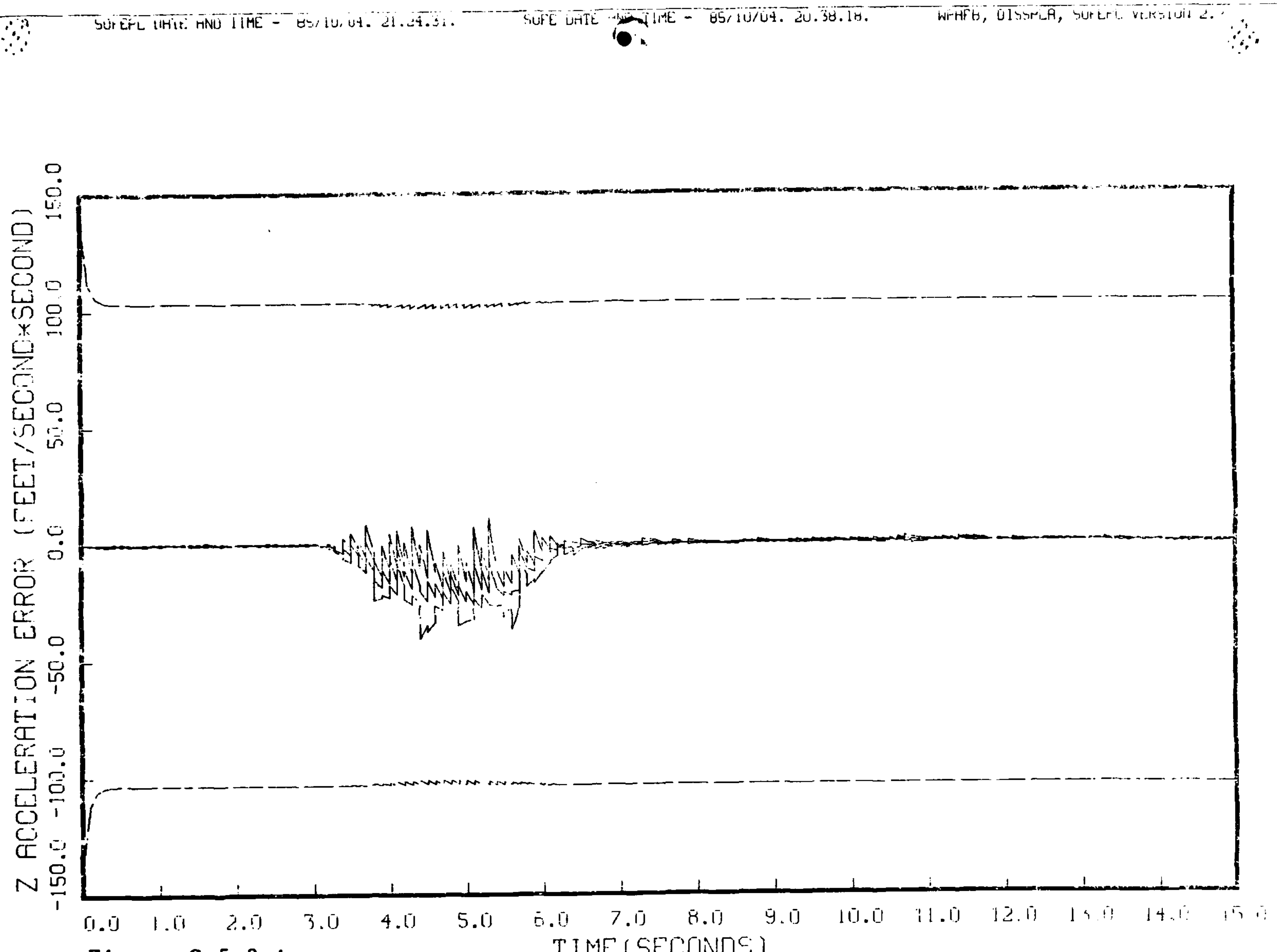


Figure G.5.3.i

STATE 9, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, R,RDOT,AZ,EL MEAS
APO 120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS

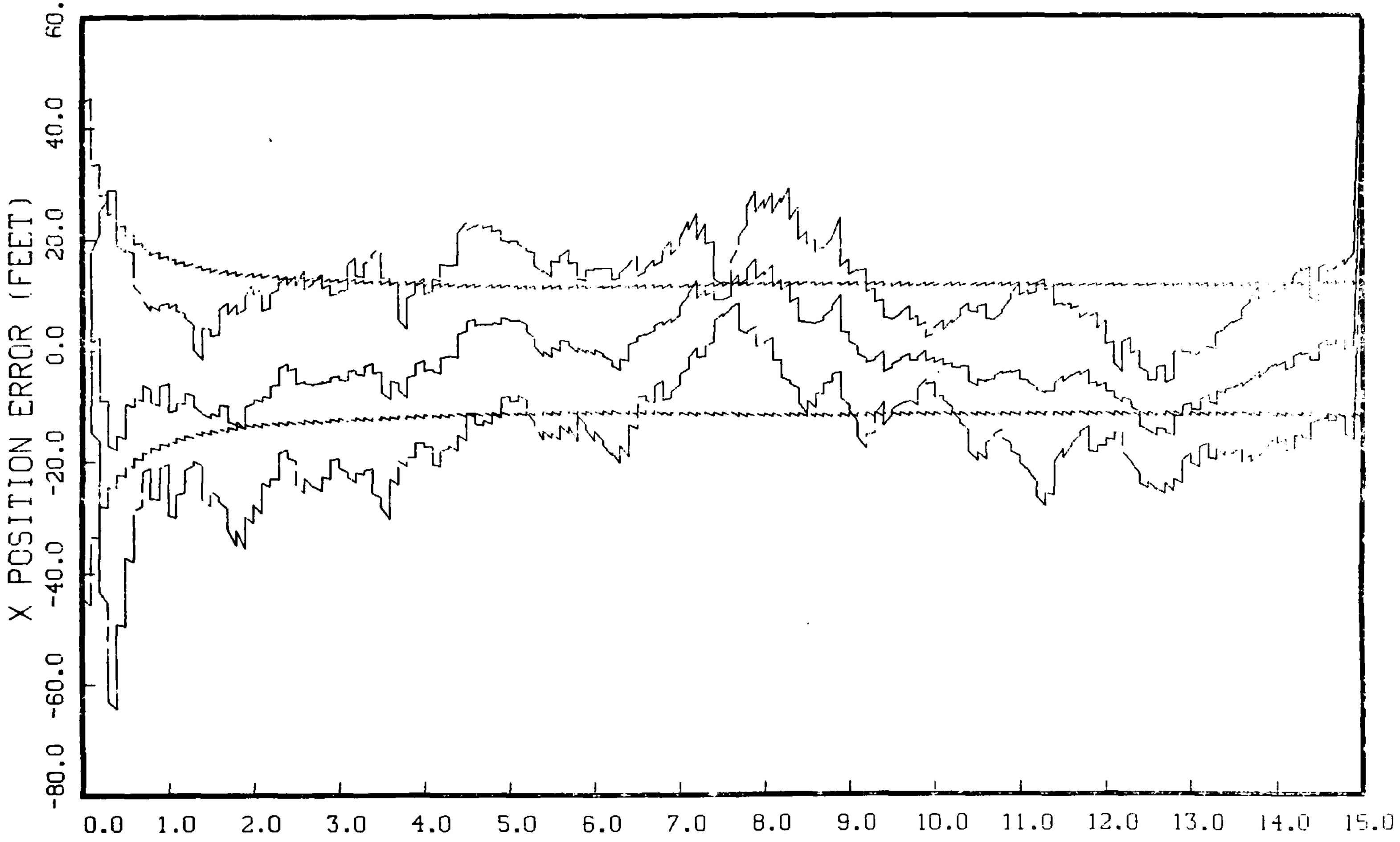


Figure G.6.1.a

STATE 1, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

ND TIME - 85/10/03. 22.45.38. SOFE DATE AND TIME - 85/10/03. 22.34.10. WPAFB, DISSPLA, SOFEPL VERSION 2.2

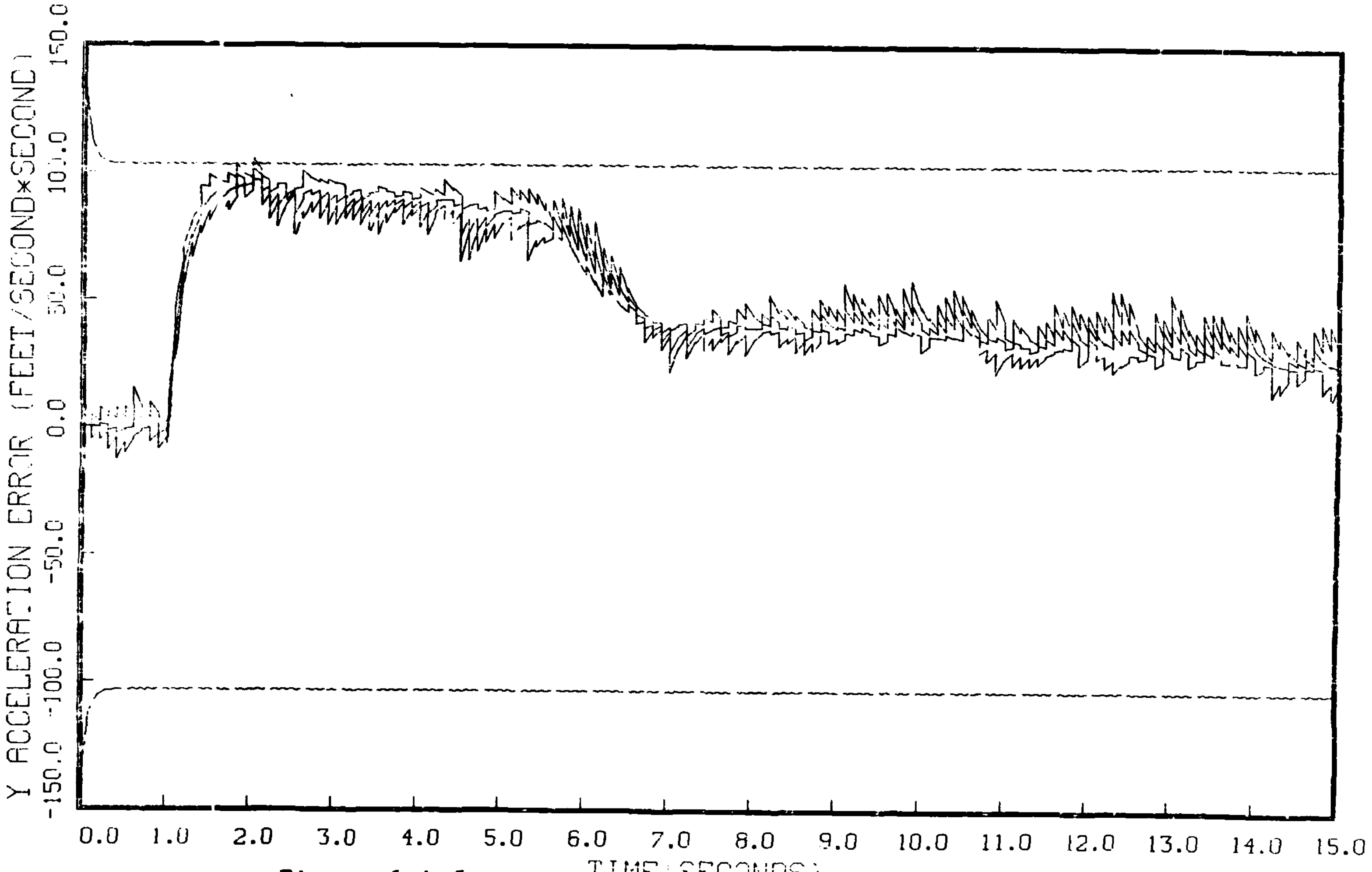


Figure 6.1.f

STATE 6, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APO-125, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

7 SOFEPL DATE AND TIME - 85/10/03. 22.45.58. SOFE DATE AND TIME - 85/10/03. 22.34.10. WPAFB, DISSPLA, SOFEPL VERSION 2.2

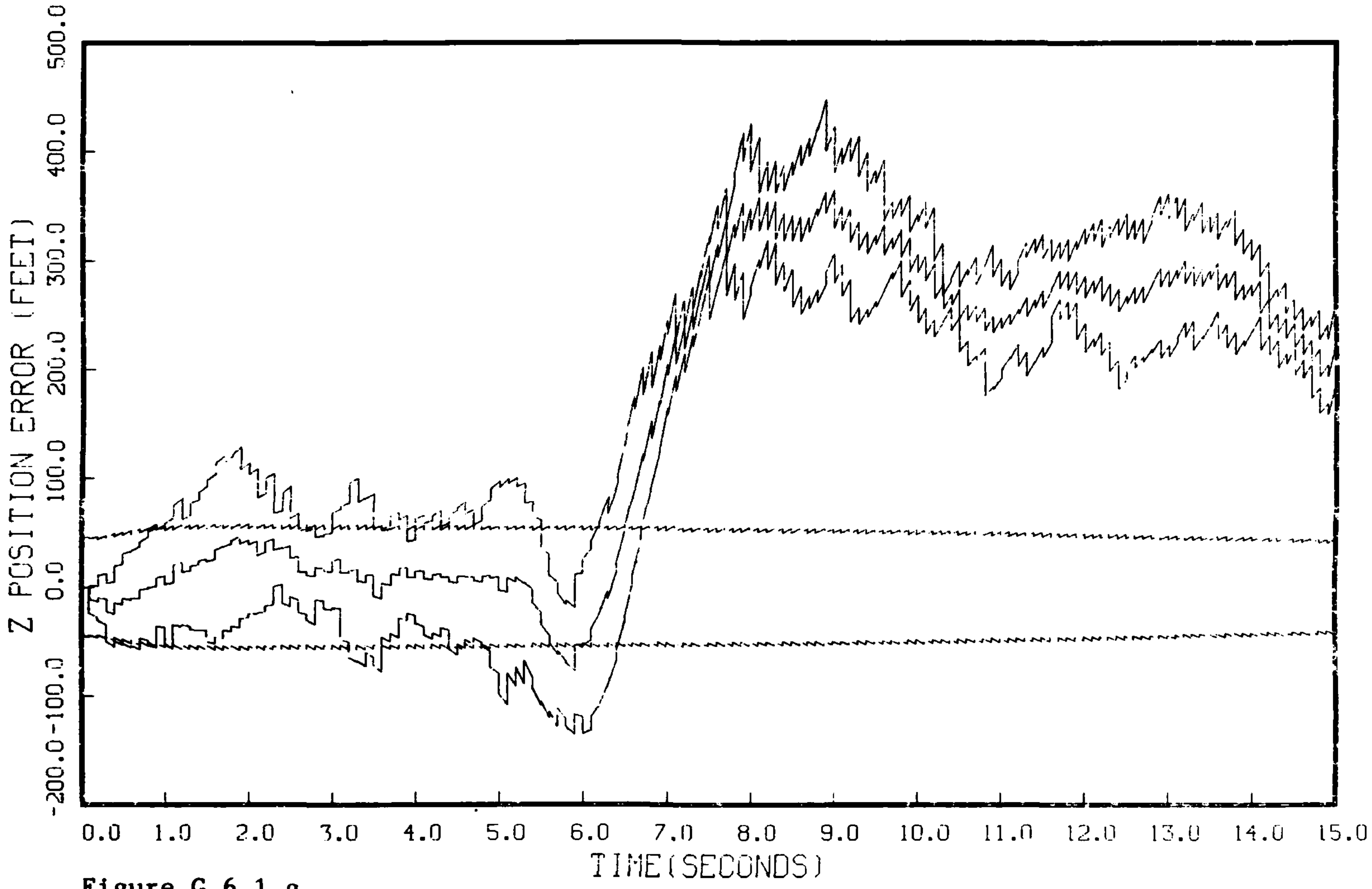


Figure G.6.1.g
STATE 7, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

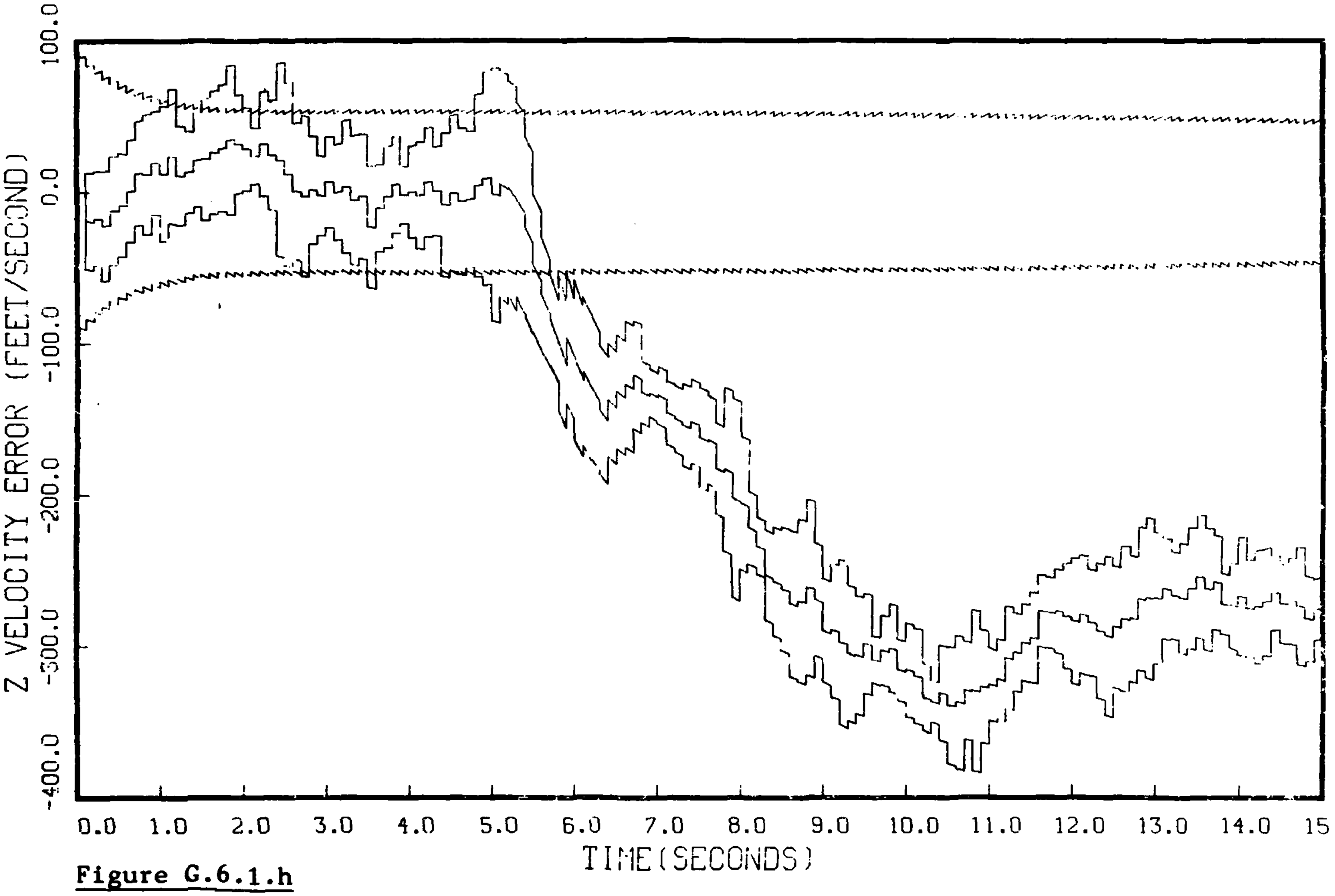


Figure G.6.1.h

STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

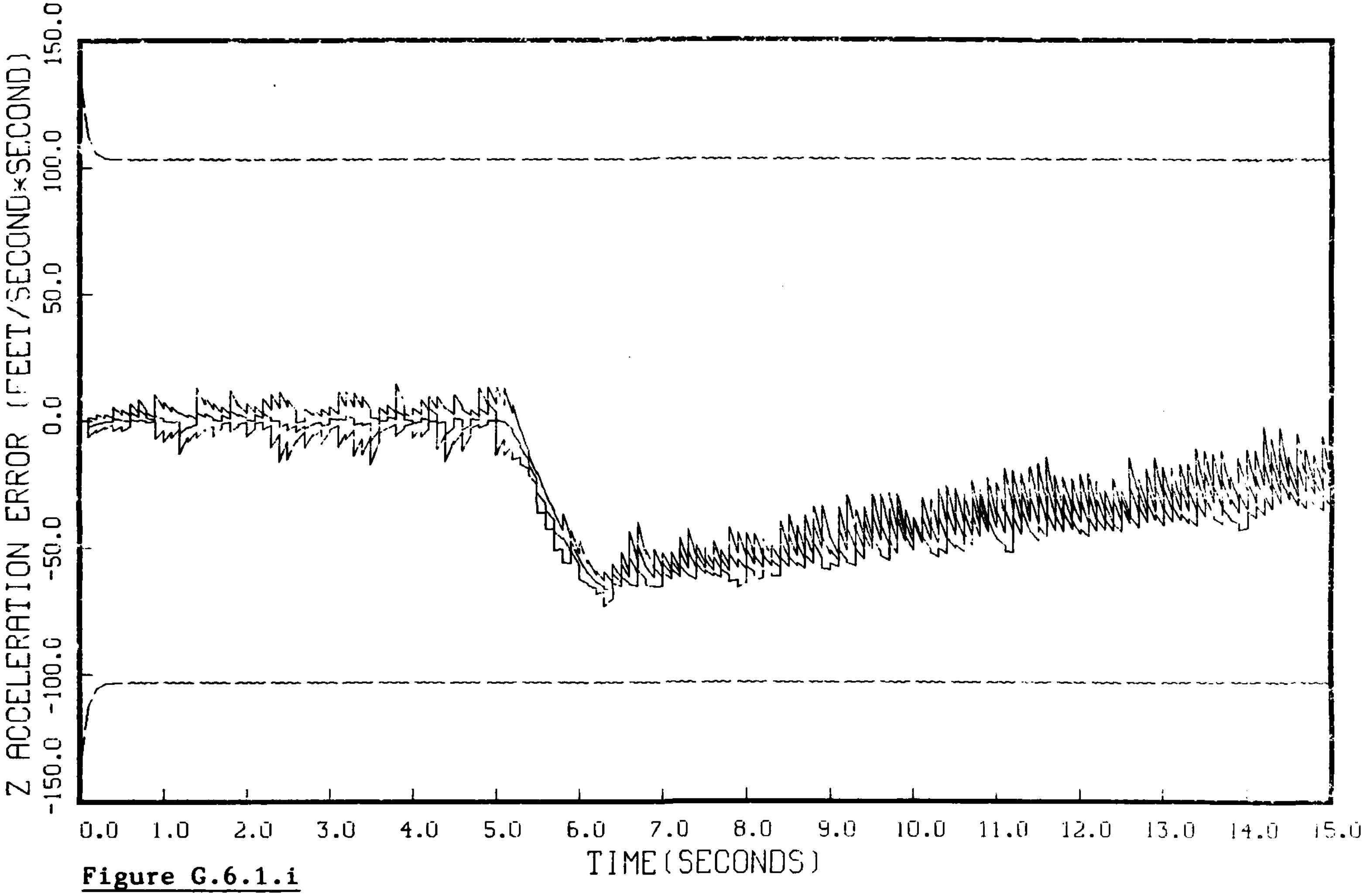


Figure G.6.1.i

STATE 9, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

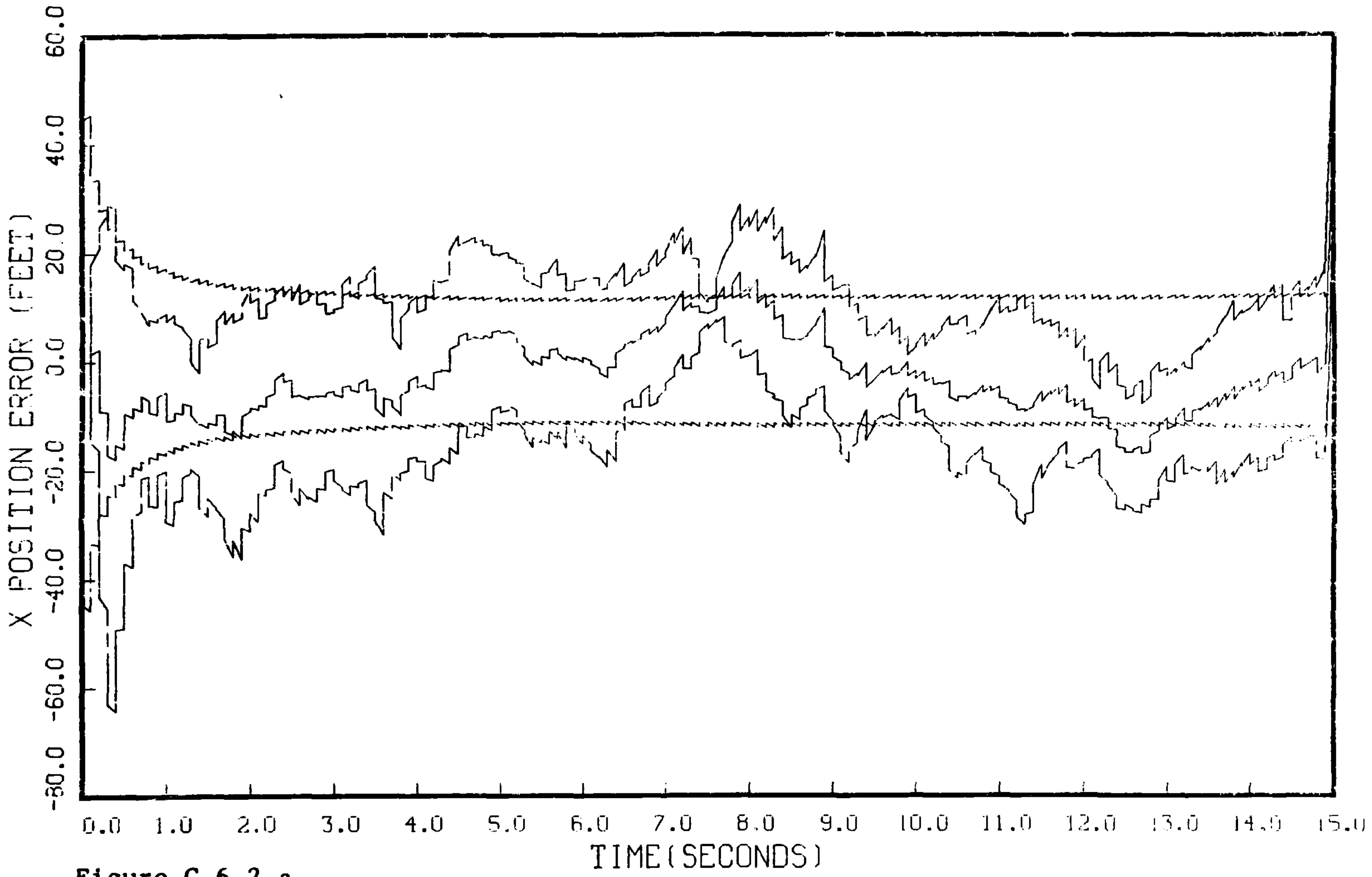


Figure G.6.2.a

STATE 1, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

SOFEPL DATE AND TIME - 85/10/04. 21.20.36. SOFE DATE AND TIME - 85/10/04. 21.06.40. WPAFB, DISSPLA, SOFEPL VERSION 2.2

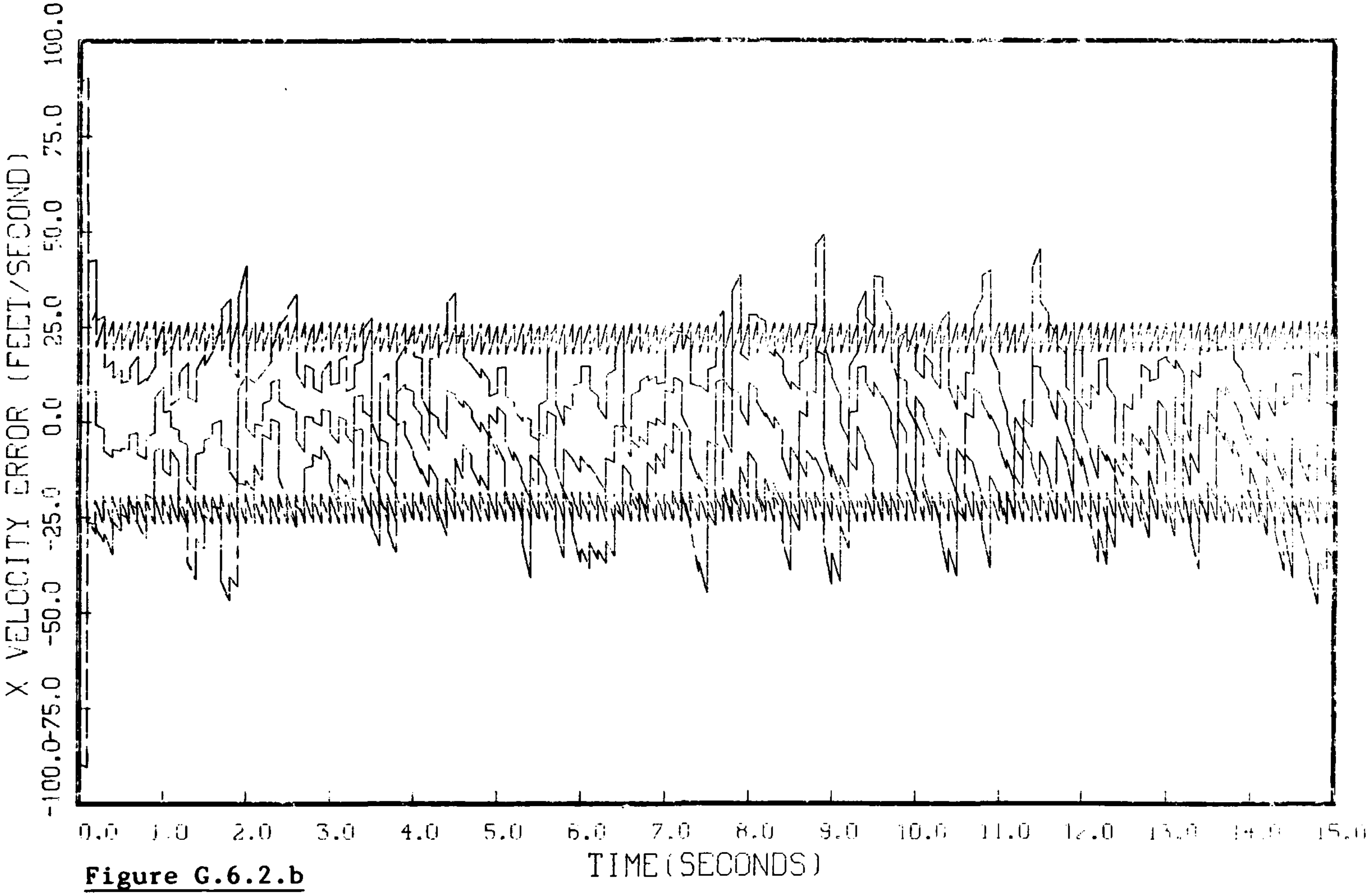


Figure G.6.2.b

STATE 2, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

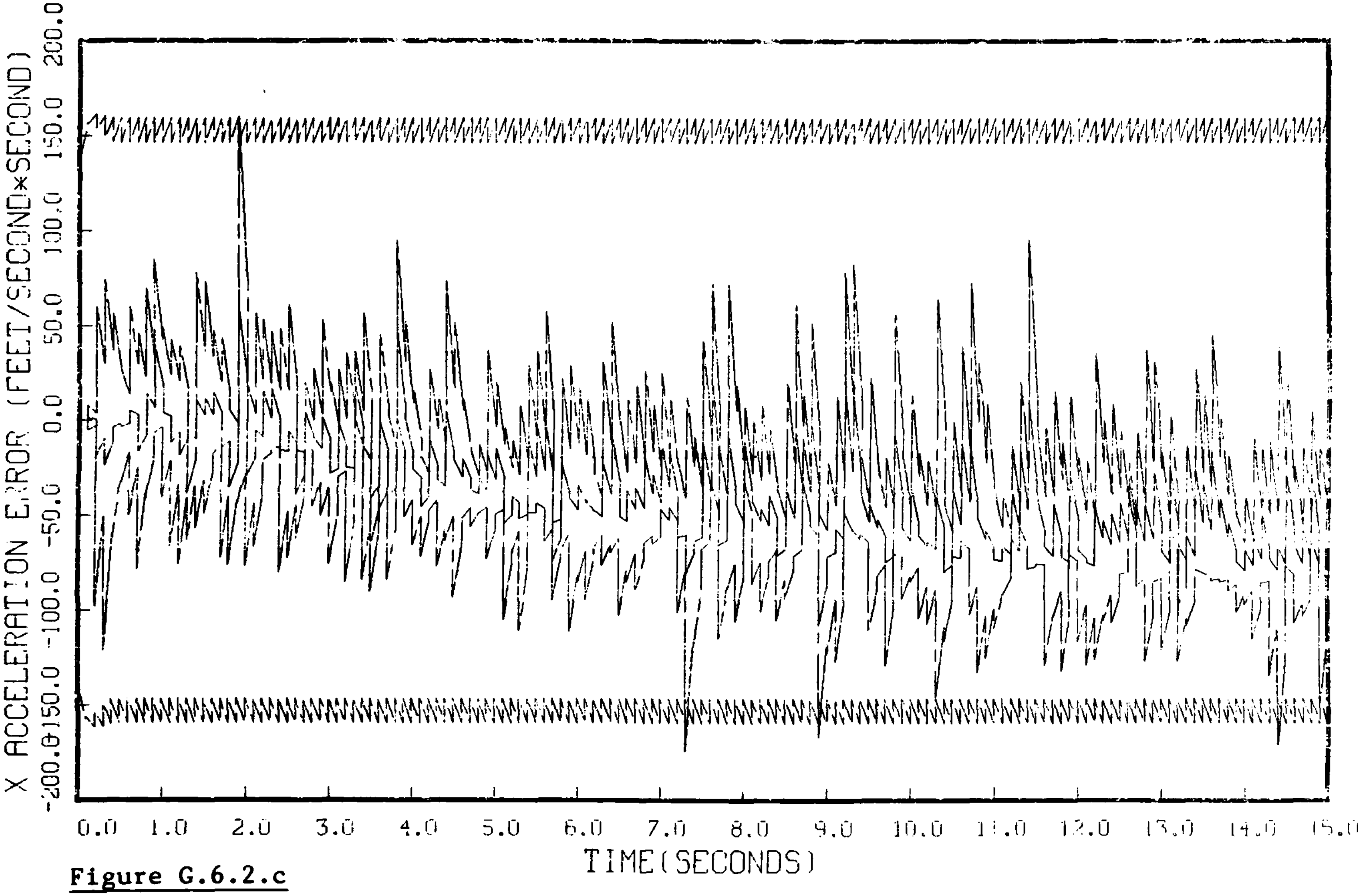


Figure G.6.2.c

STATE 3, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

SOFEPL DATE AND TIME - 85/10/04. 21.20.36.   SOFE DATE AND TIME - 85/10/04. 21.06.40.   WPAFB, DISSPLA, SOFEPL VERSION 2.3

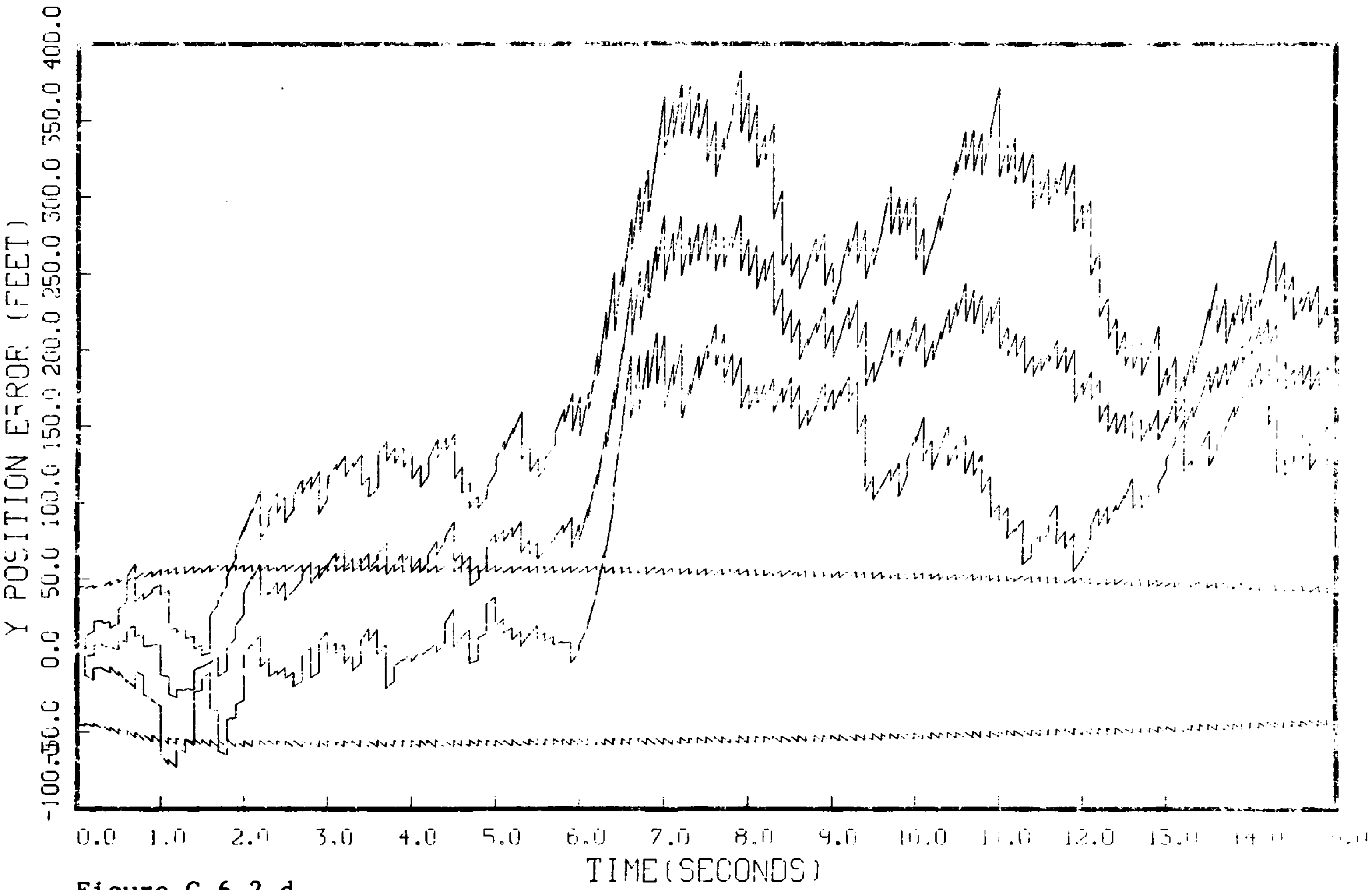


Figure G.6.2.d

STATE 4, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
MPC 125, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

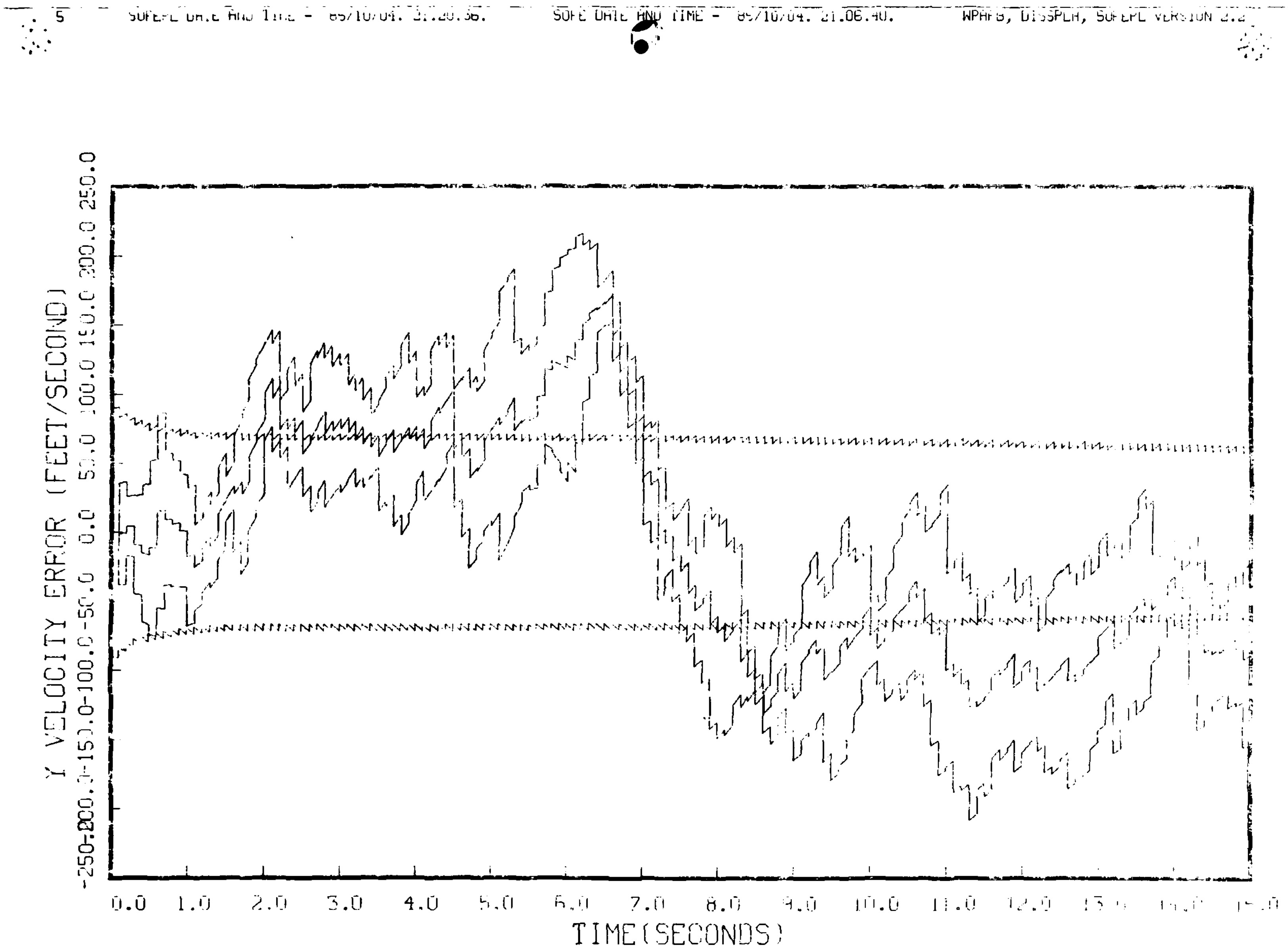


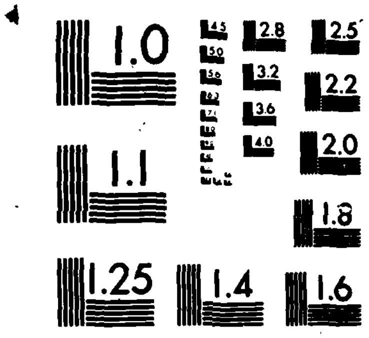
Figure G.6.2.e
STATE 5, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
HPO 120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

AD-A164 034 PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT 5/5
MANEUVERING TARGET..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R B ANDERSON
UNCLASSIFIED DEC 85 AFIT/GE/ENG/85D-2 F/G 19/5 NL

END

FILMED

DTIC


1.0
4.5
5.0
5.6
6.3
7.1
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

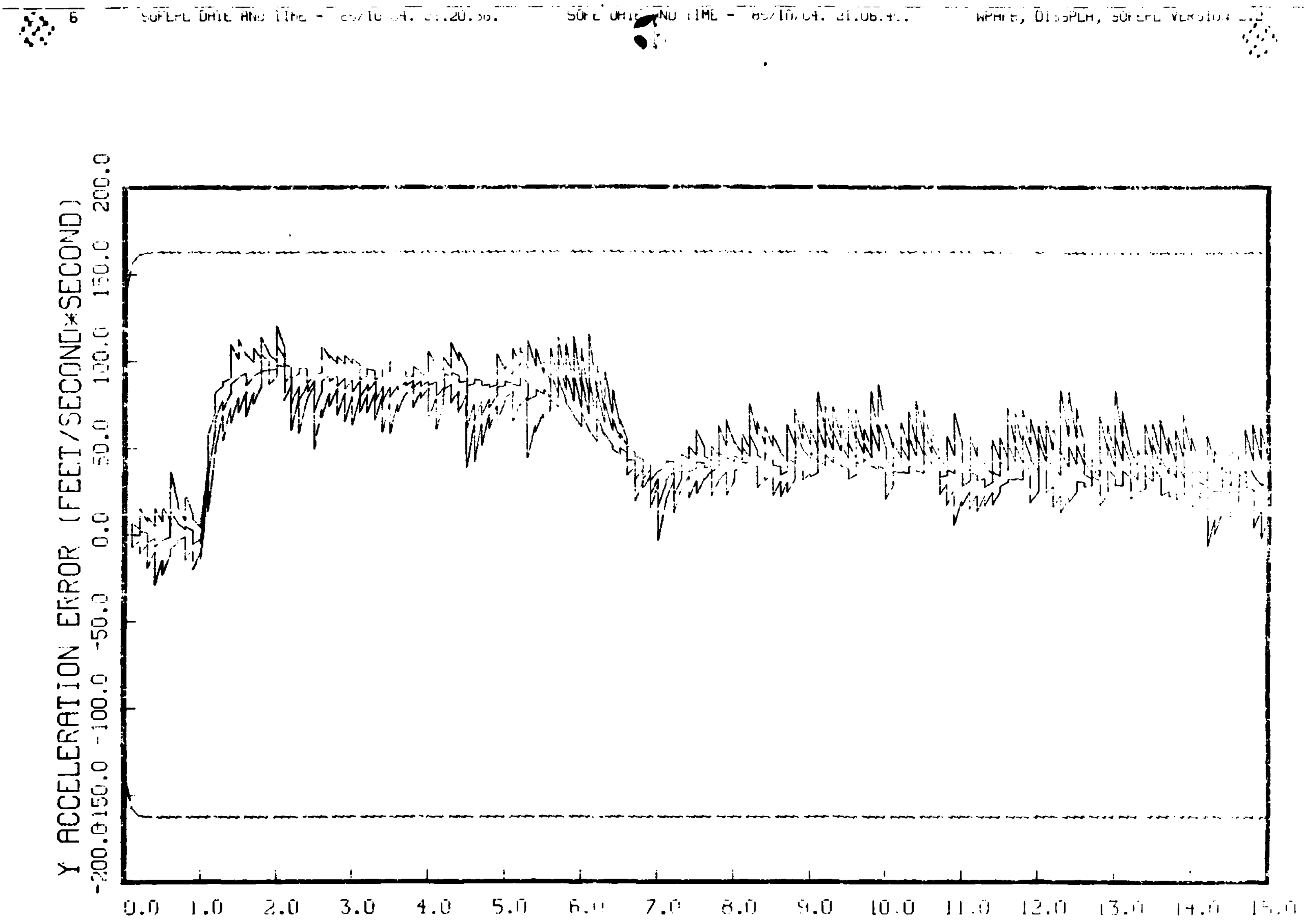


Figure G.6.2.f
STATE 6, Q(1)-Q(2)-Q(3)-373250., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APG-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

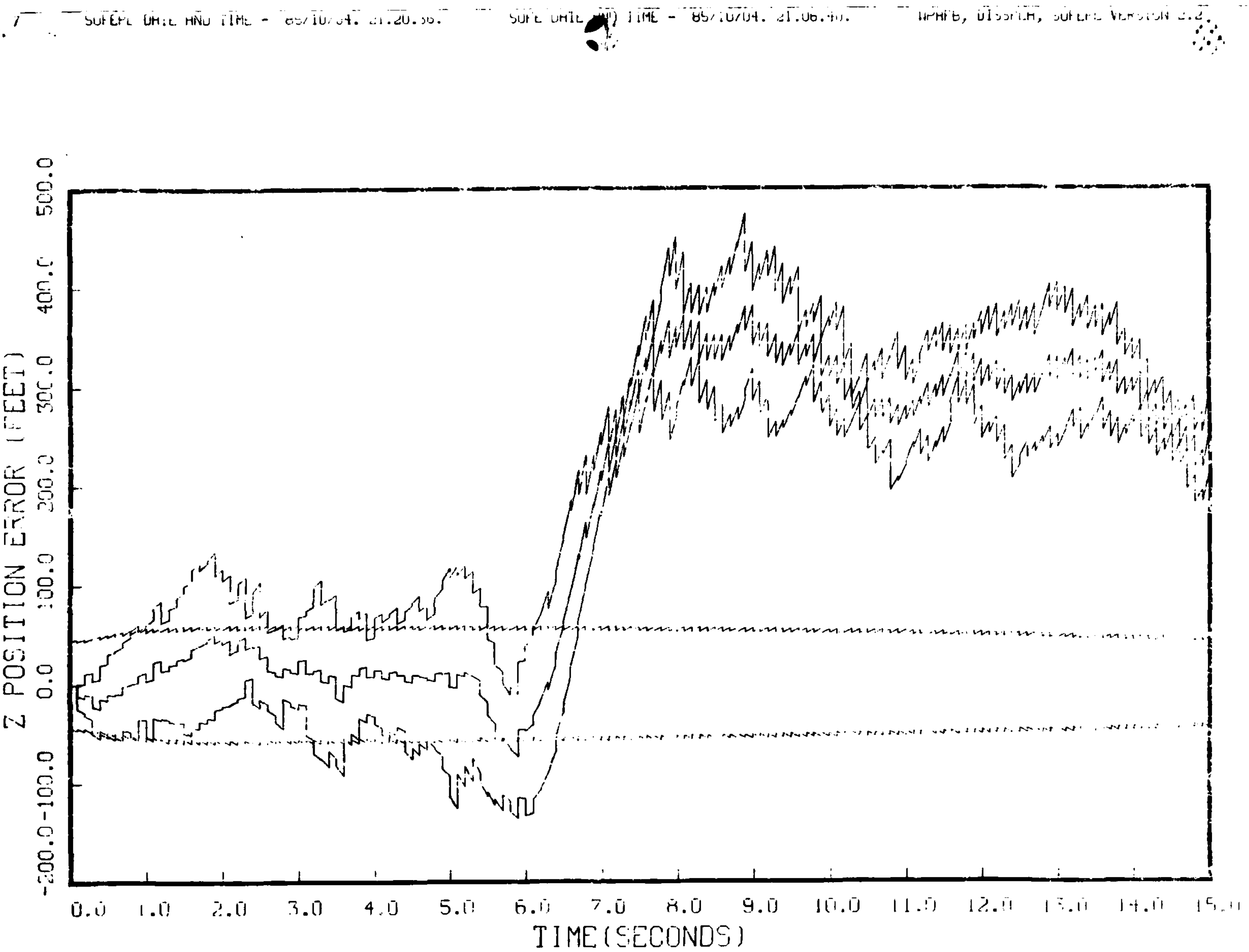


Figure G.6.2.g
STATE 7, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ 120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

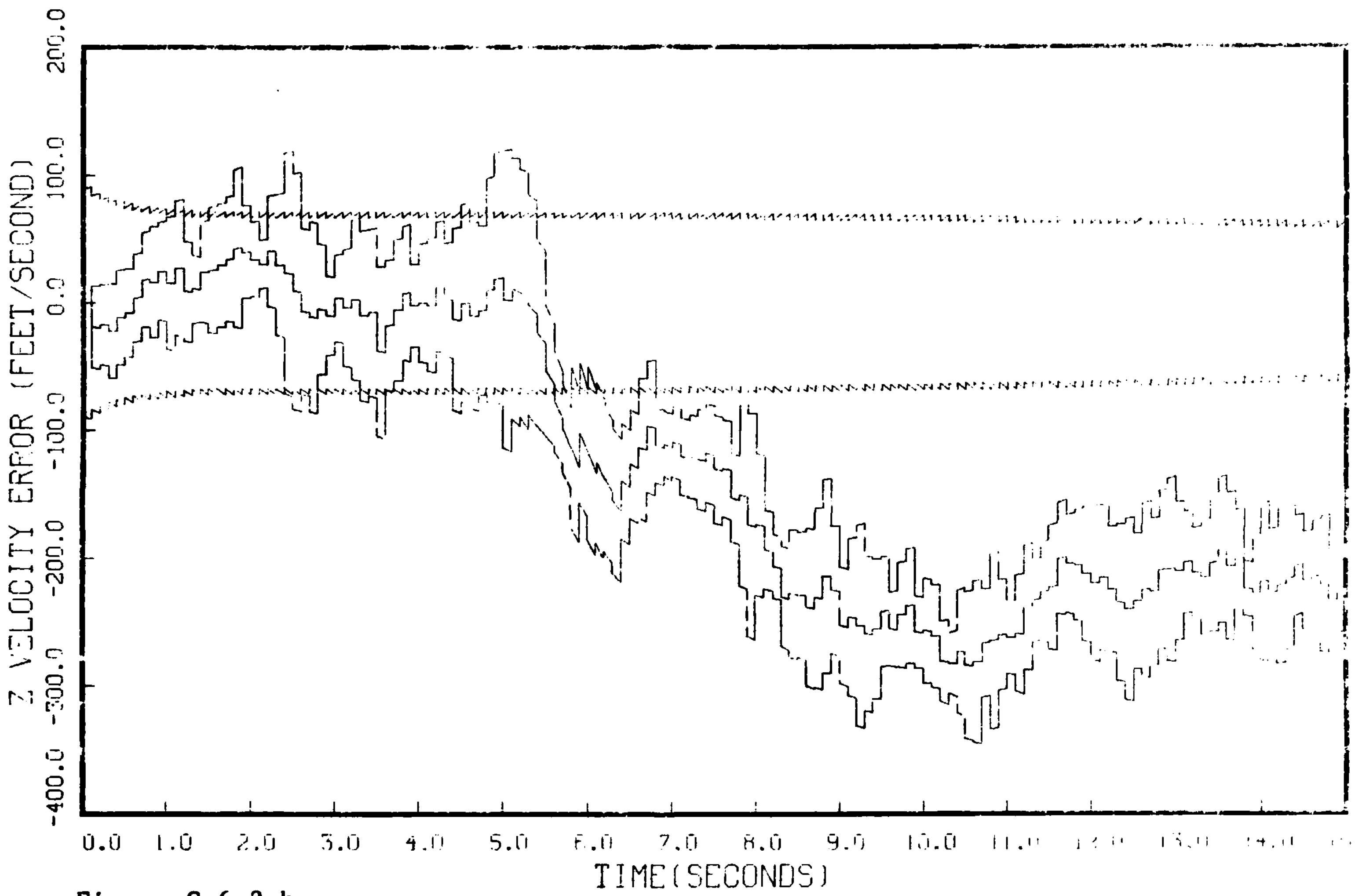


Figure G.6.2.h
STATE 8, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, TAIL CHASE, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS

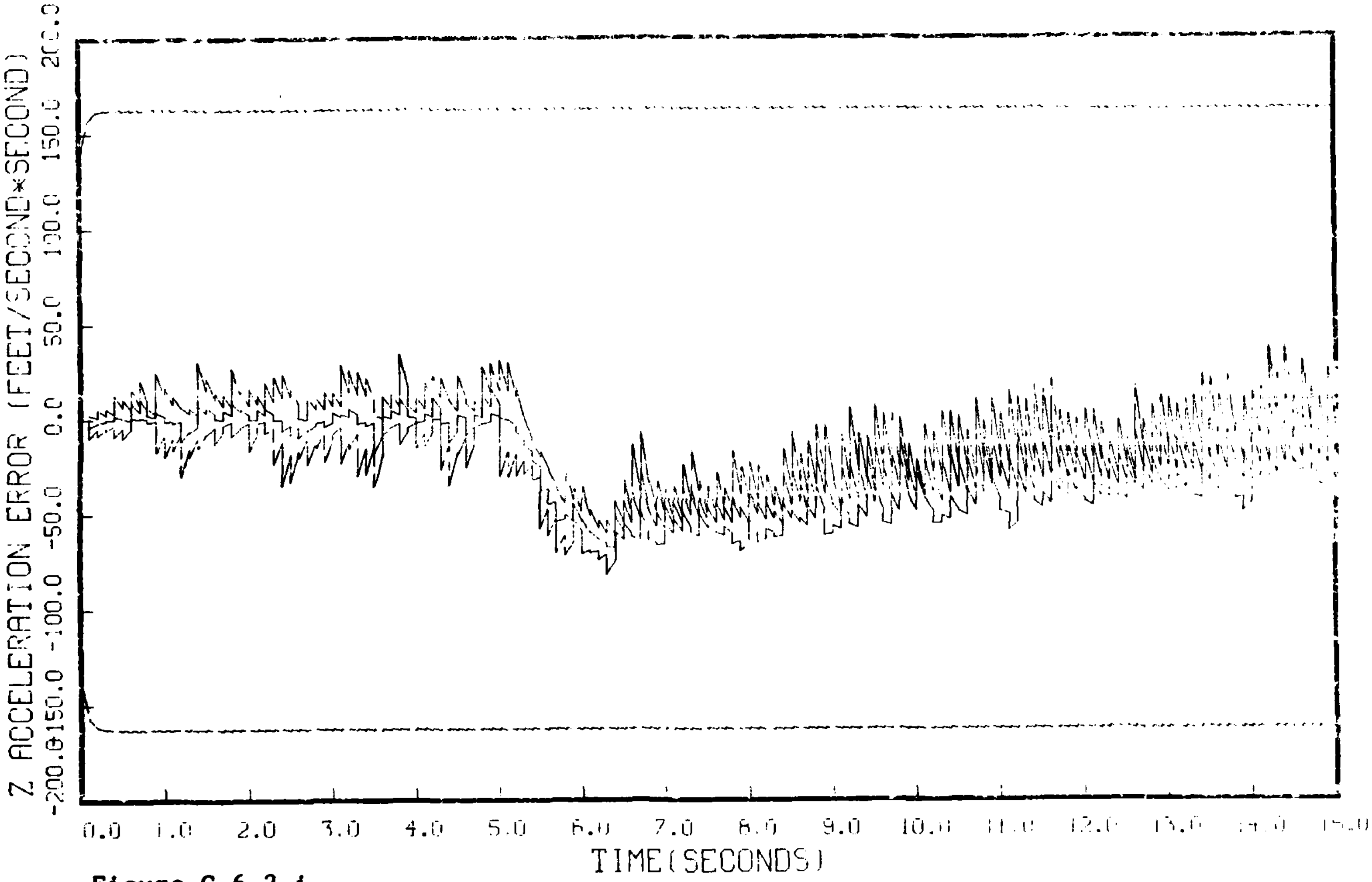


Figure G.6.2.i
STATE 9, Q(1)=Q(2)=Q(3)=373250., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
HPO=120, TAIL CHASE, INITIAL RANGE=15,000.', UPDATE=.1, 5 RUNS

8A PS 6 1 SOFEPL DATE AND TIME - 85/10/13. 10.24.30. SOFE DATE AND TIME - 85/10/13. 10.18.36. WPAFB, DISSPLA, SOFEPL VERSION 2.2

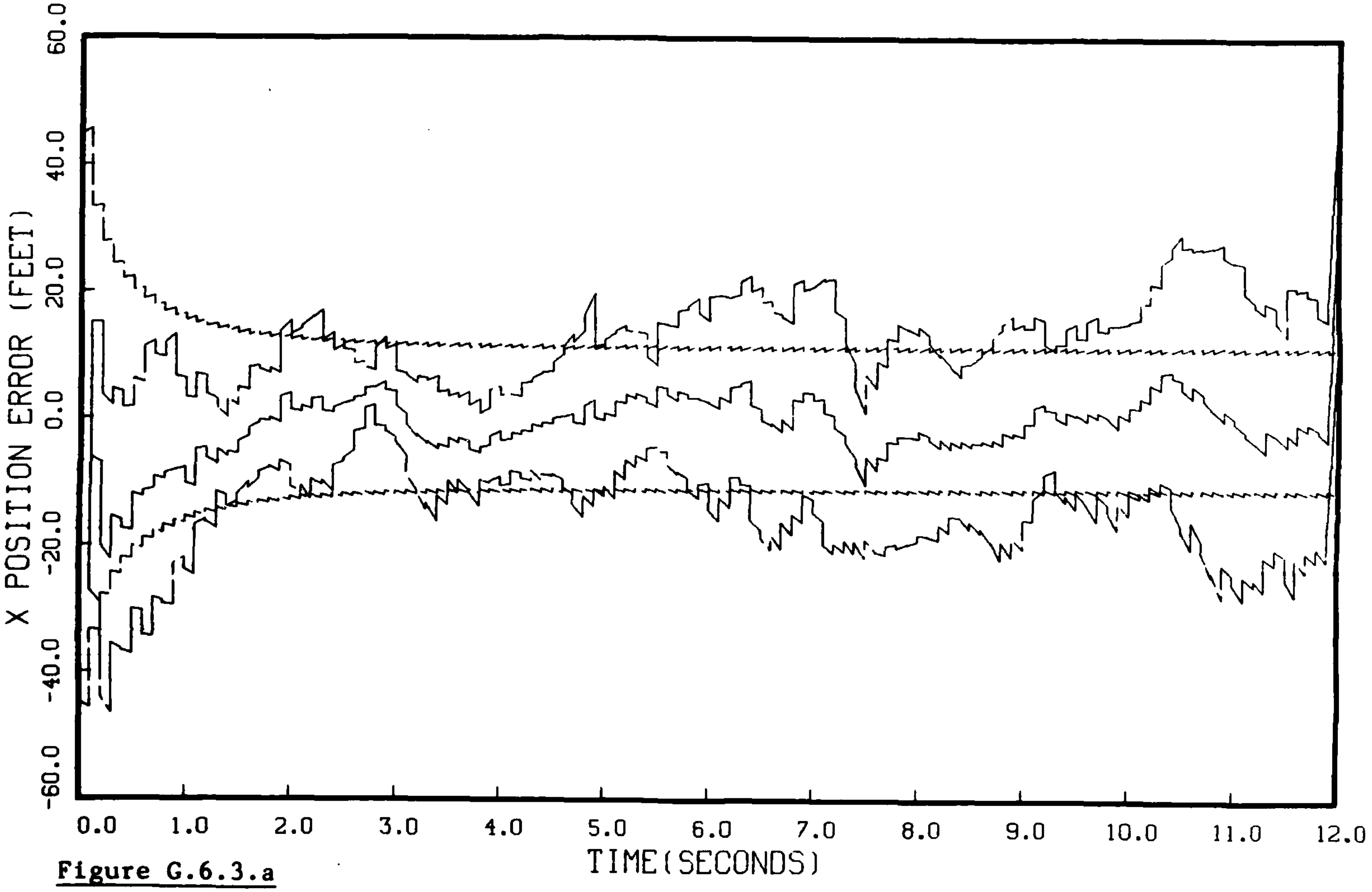


Figure G.6.3.a

STATE 1, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

2 SOFEPL DATE AND TIME - 85/10/13. 10.24.30. SOFE DATE AND TIME - 85/10/13. 10.18.36. WPAFB, DISSPLA, SOFEPL VERSION 2.2

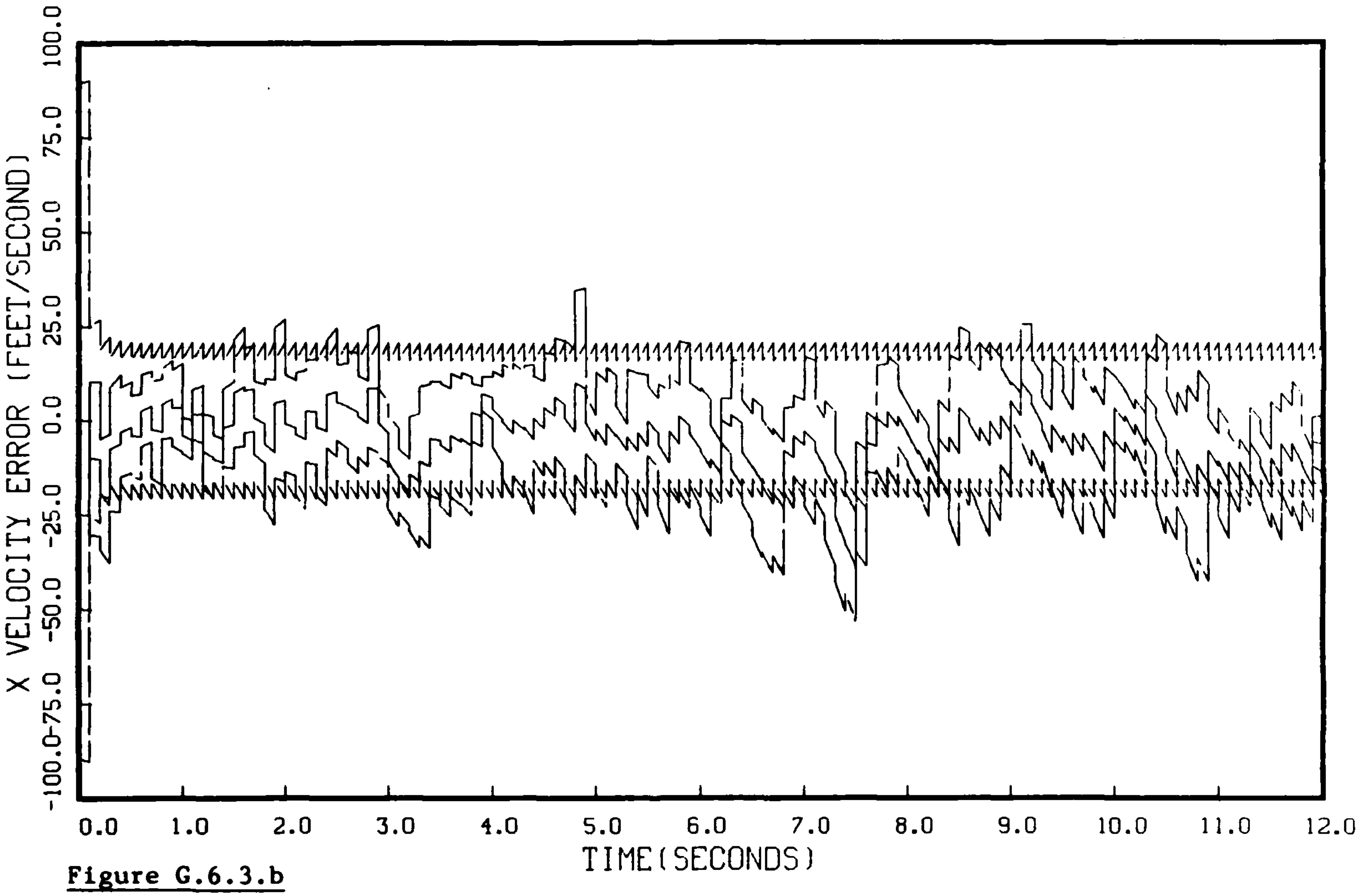


Figure G.6.3.b

STATE 2, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE-10,000. , UPDATE-.1, 5 RUNS

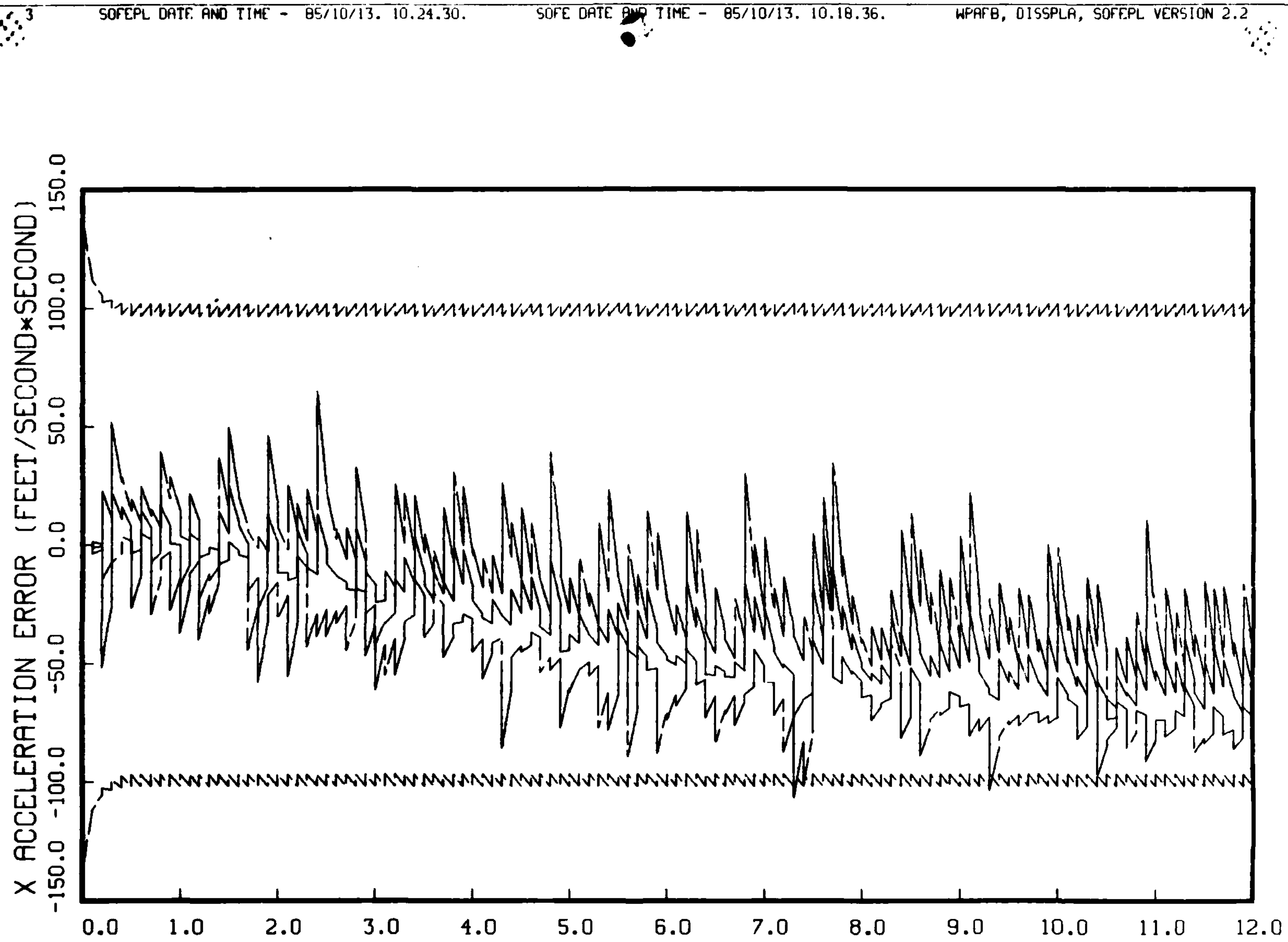


Figure G.6.3.c

STATE 3, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

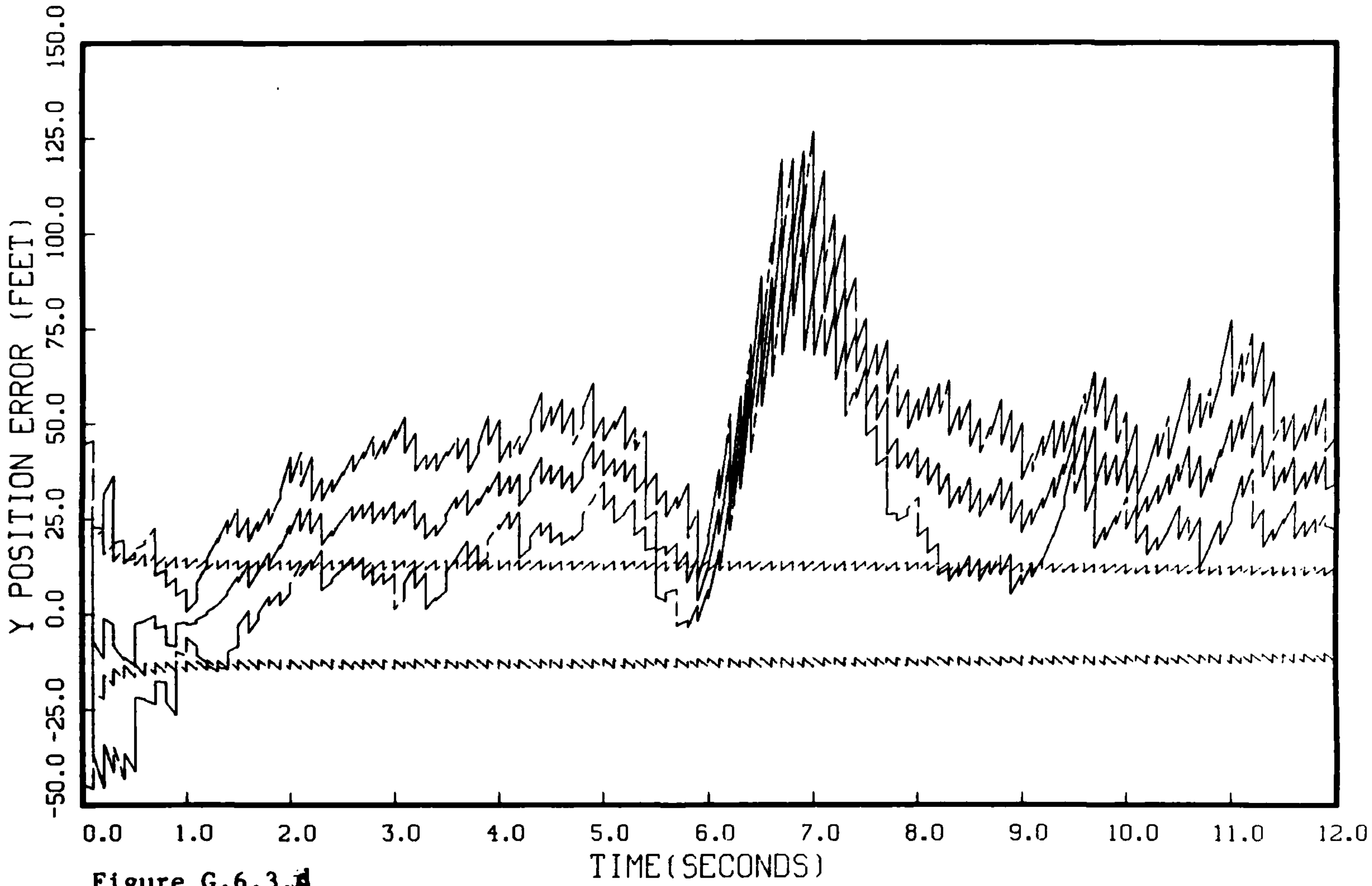


Figure G.6.3.d

STATE 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

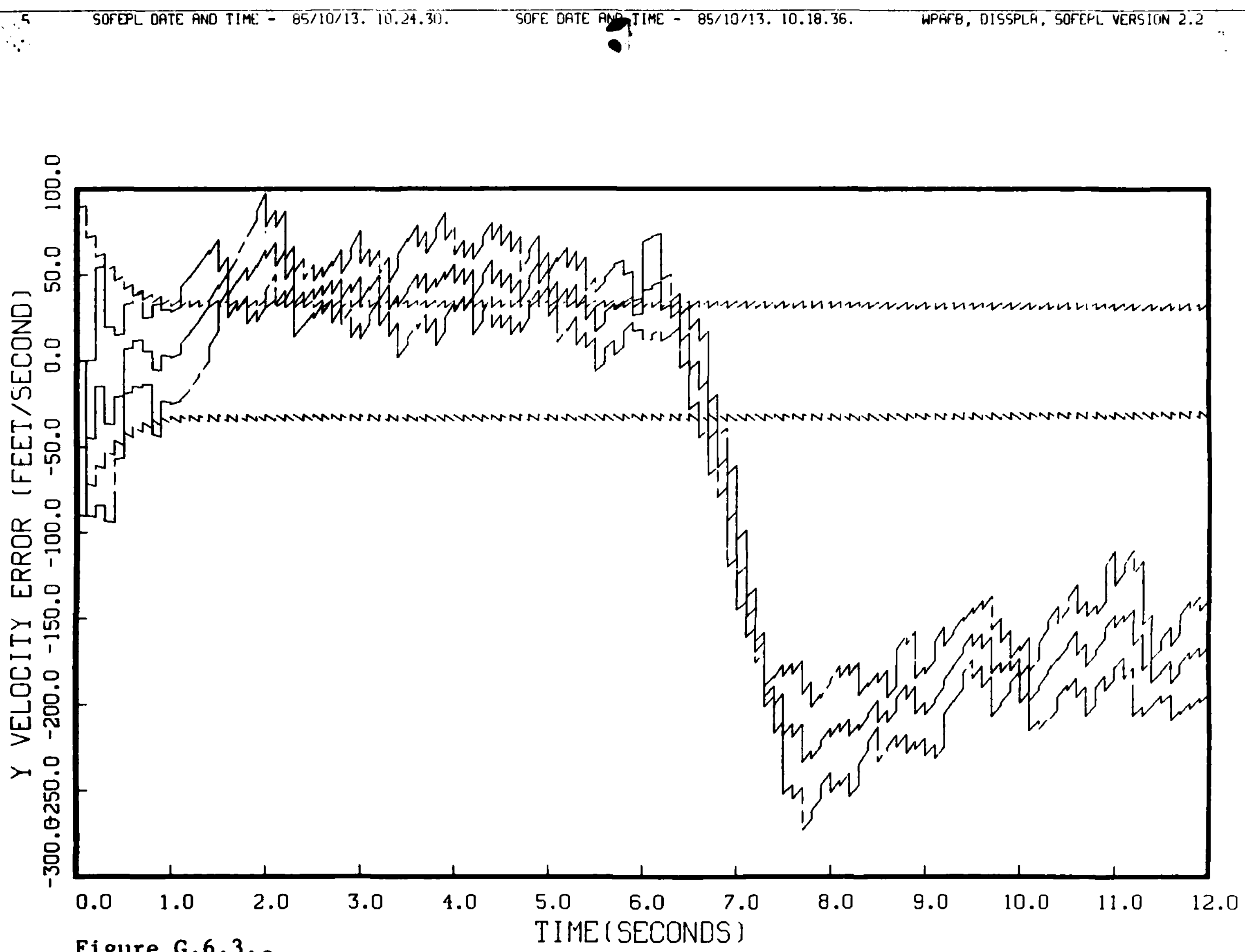


Figure G.6.3.e

STATE 5, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

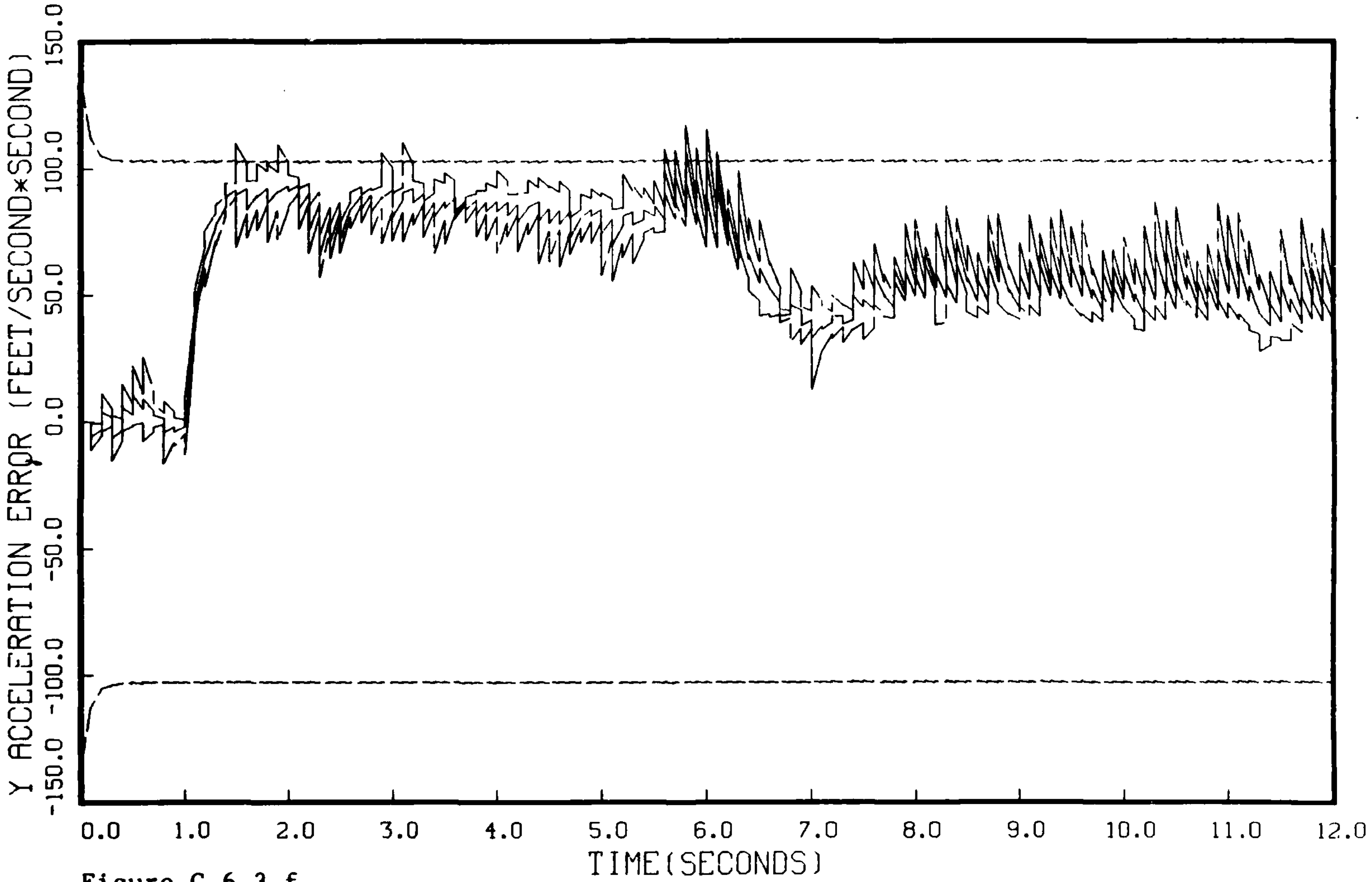


Figure G.6.3.f

STATE 6, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

SOFEPL DATE AND TIME - 85/10/13. 10.24.30. SOFE DATE AND TIME - 85/10/13. 10.18.36. WPAFB, DISSPLA, SOFEPL VERSION 2.2

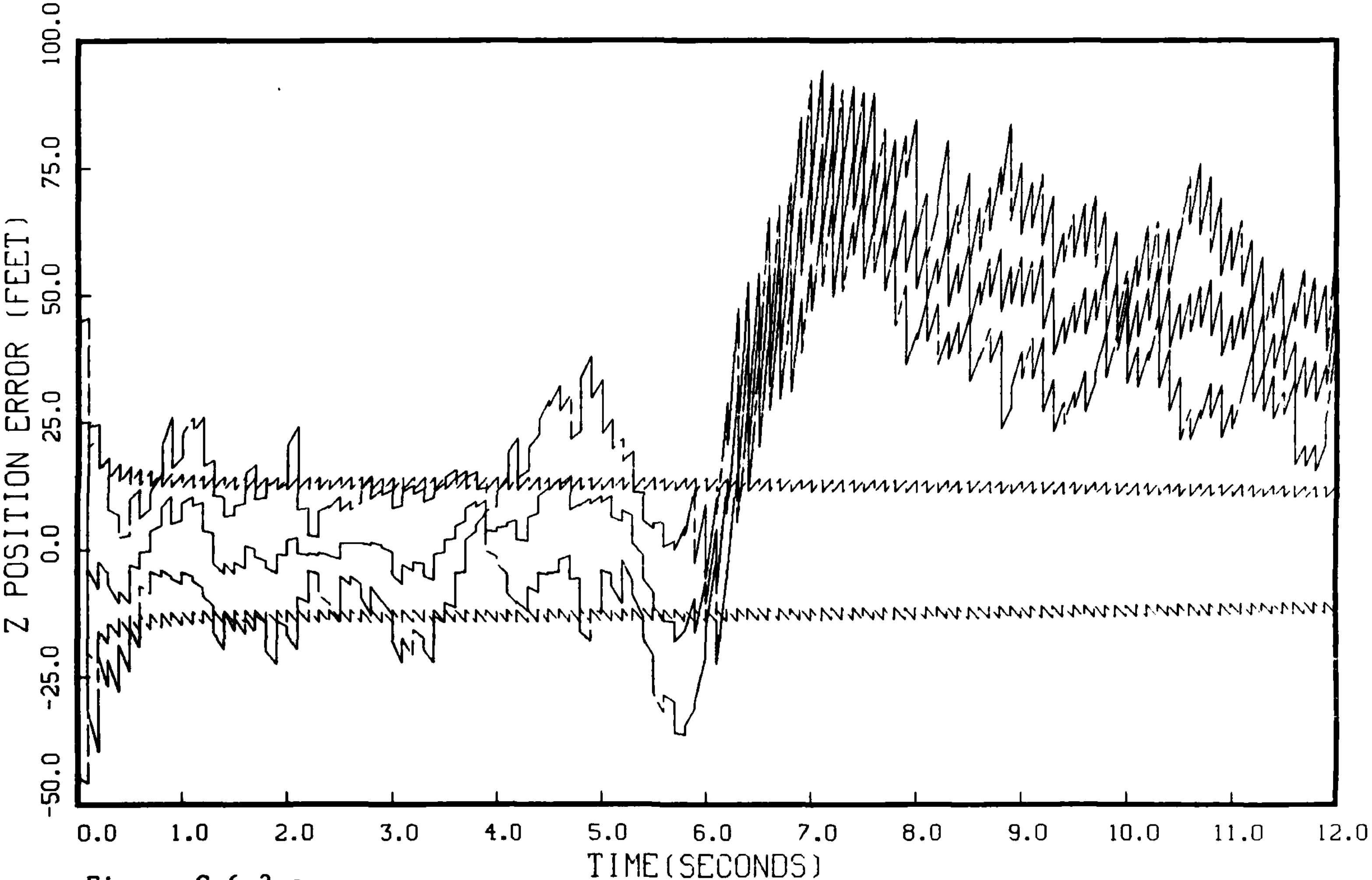


Figure G.6.3.g

STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

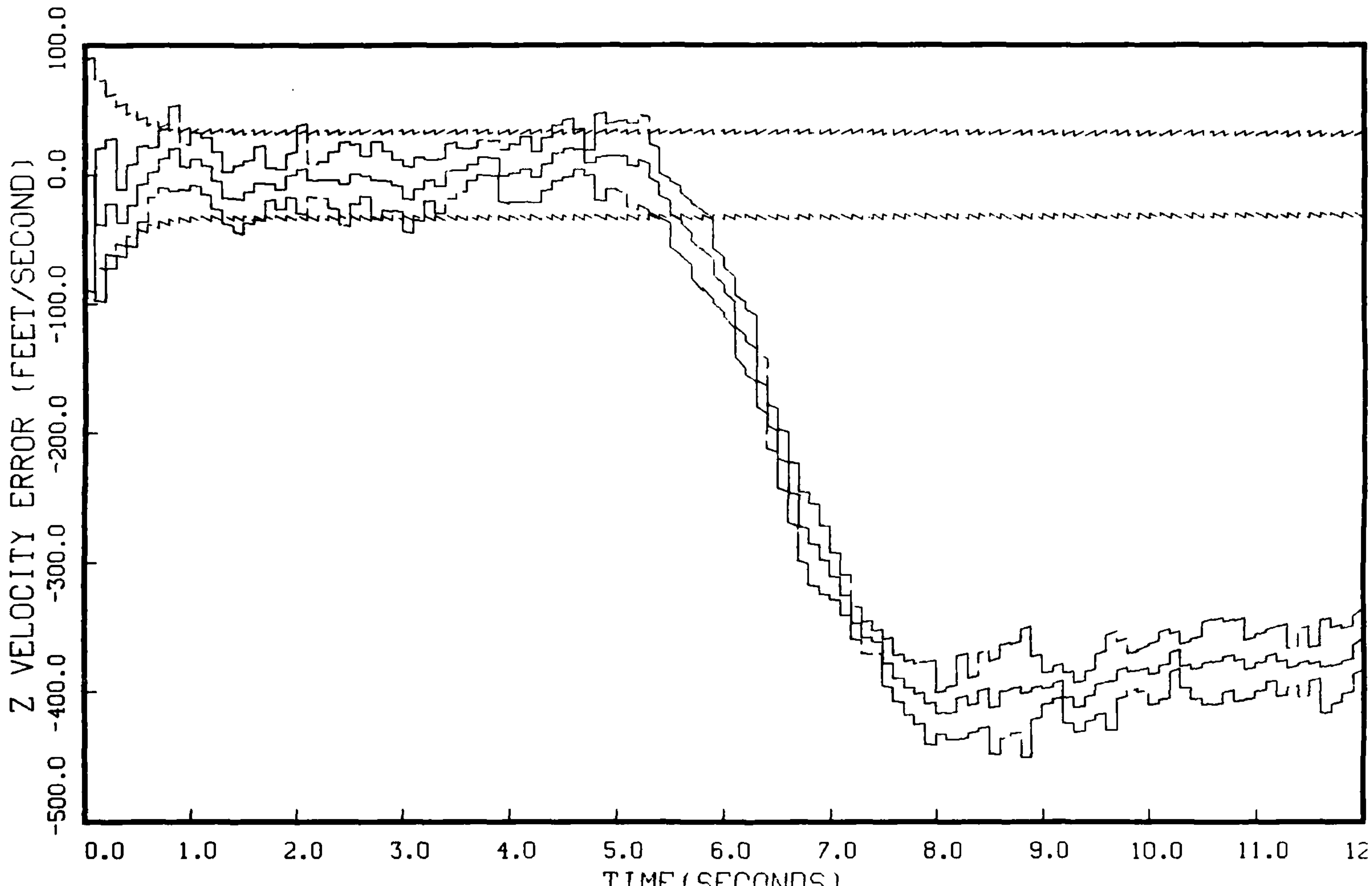


Figure G.6.3.h

STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, TAIL CHASE, INITIAL RANGE-10,000.', UPDATE-.1, 5 RUNS

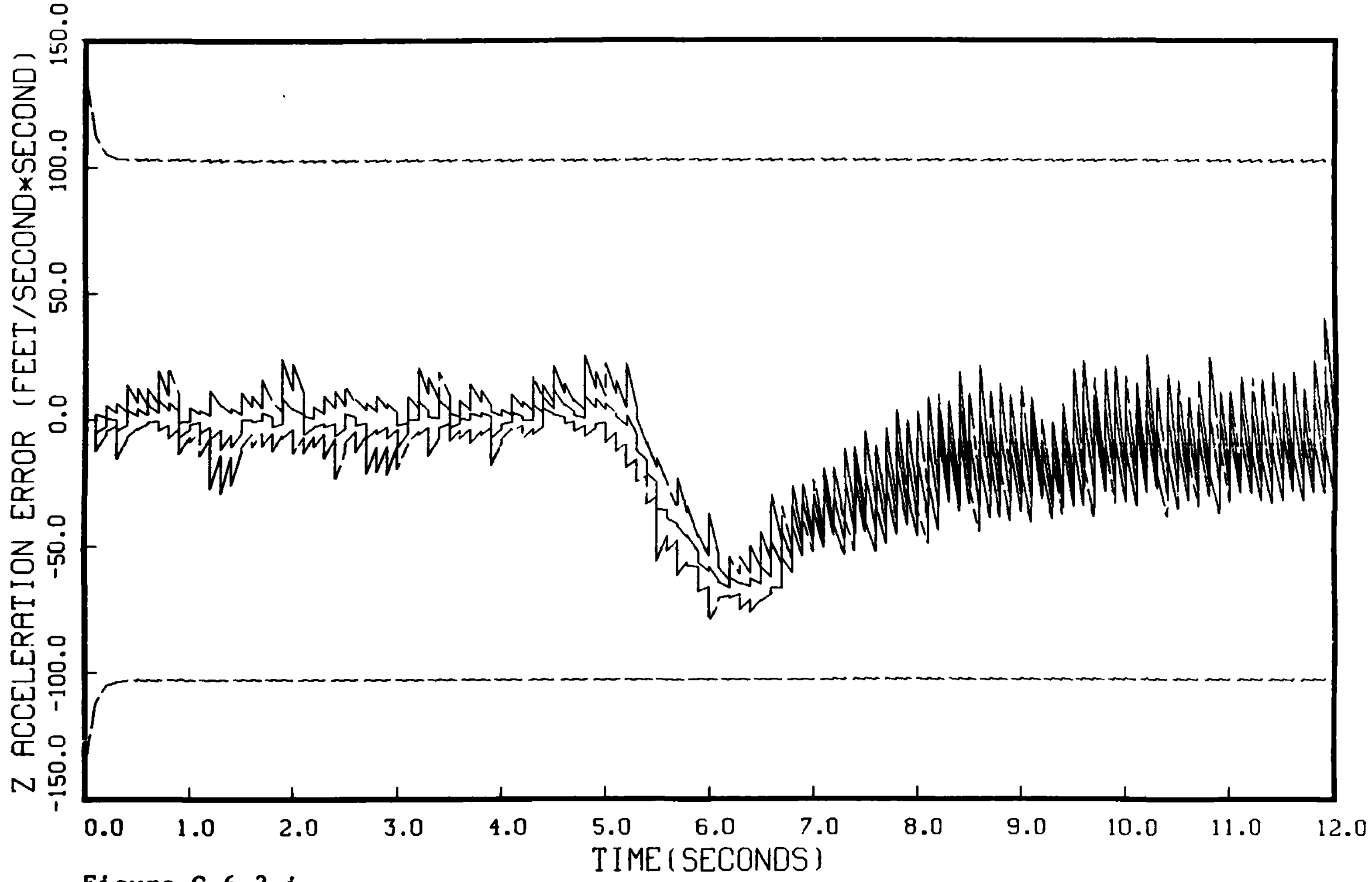
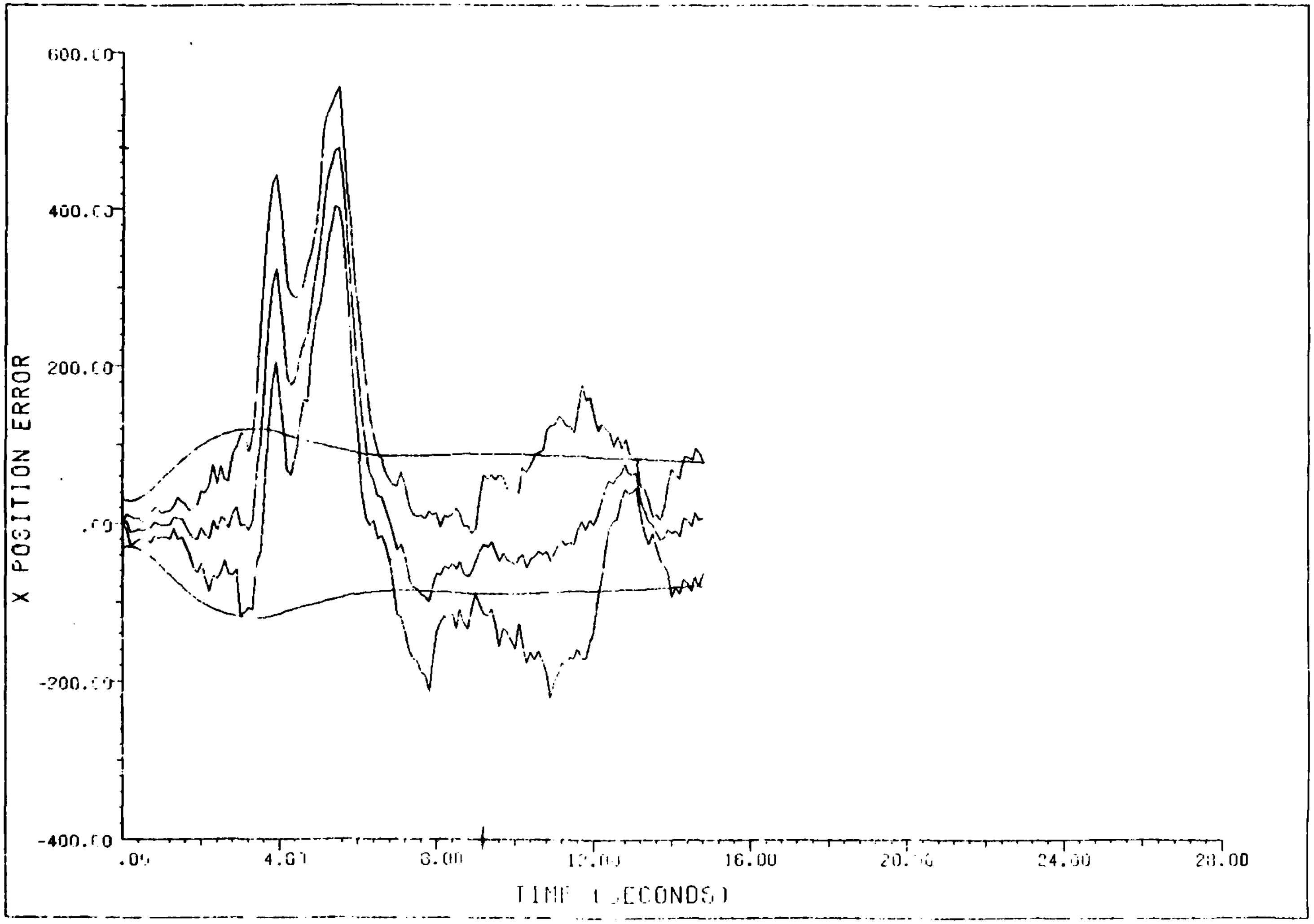


Figure G.6.3.i
STATE 9, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, TAIL CHASE, INITIAL RANGE-10,000. , UPDATE-.1, 5 RUNS

Figure Set G.7.1 is the same as Figure Set G.3.1



APO120 Q = VAR TAU=.143 6 MEAS. 5 RUNS

Figure G.7.2.a

X

X velocity

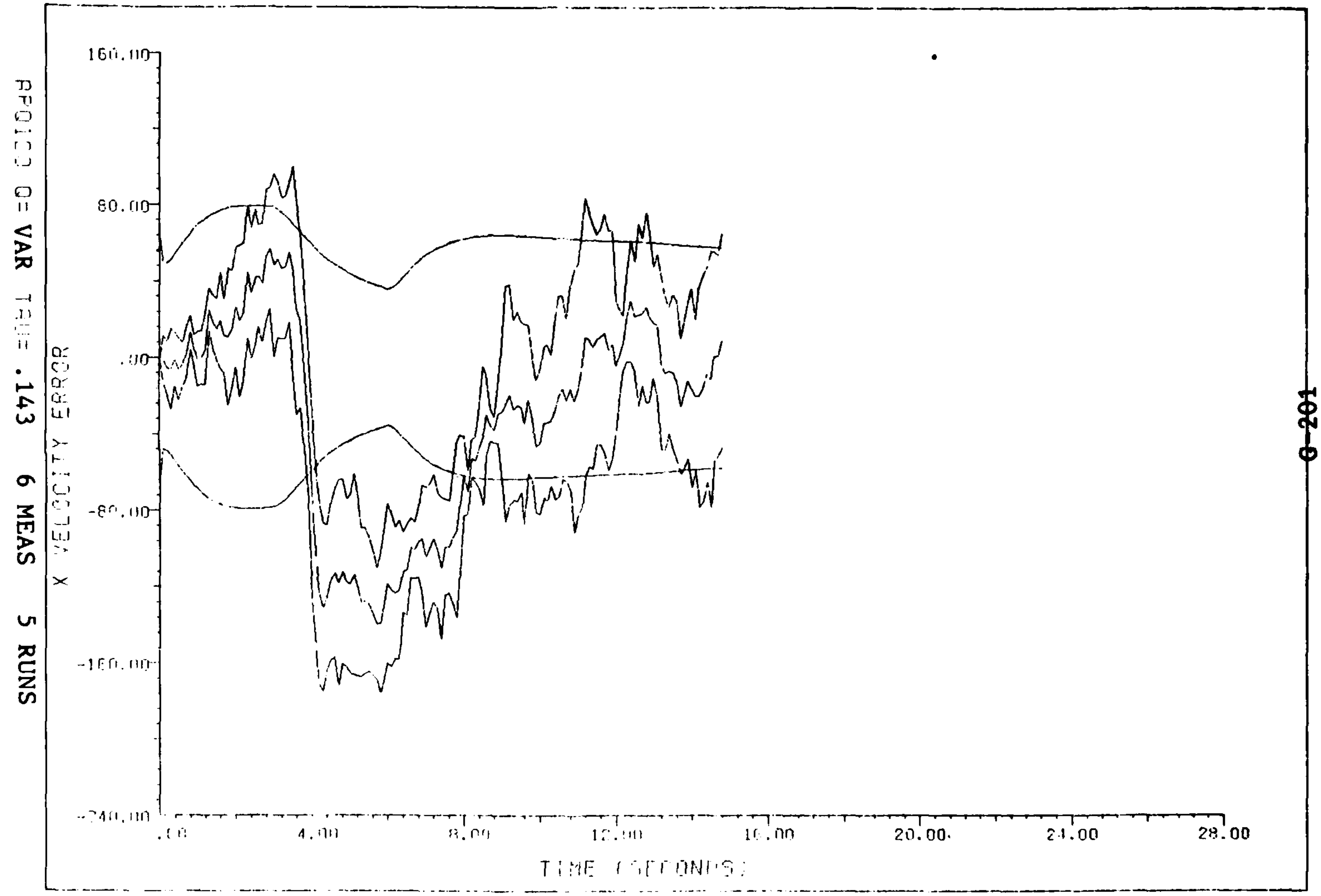
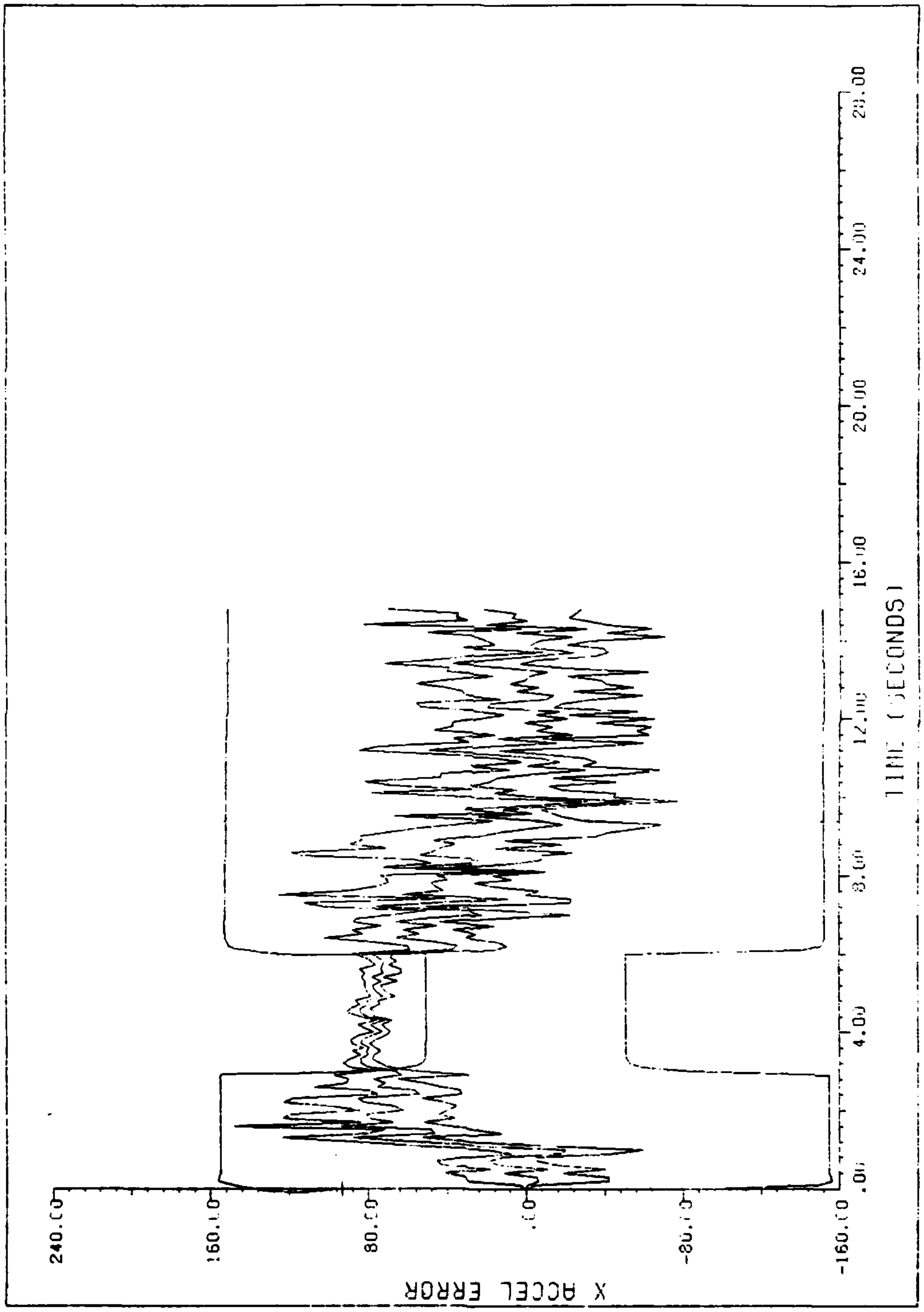
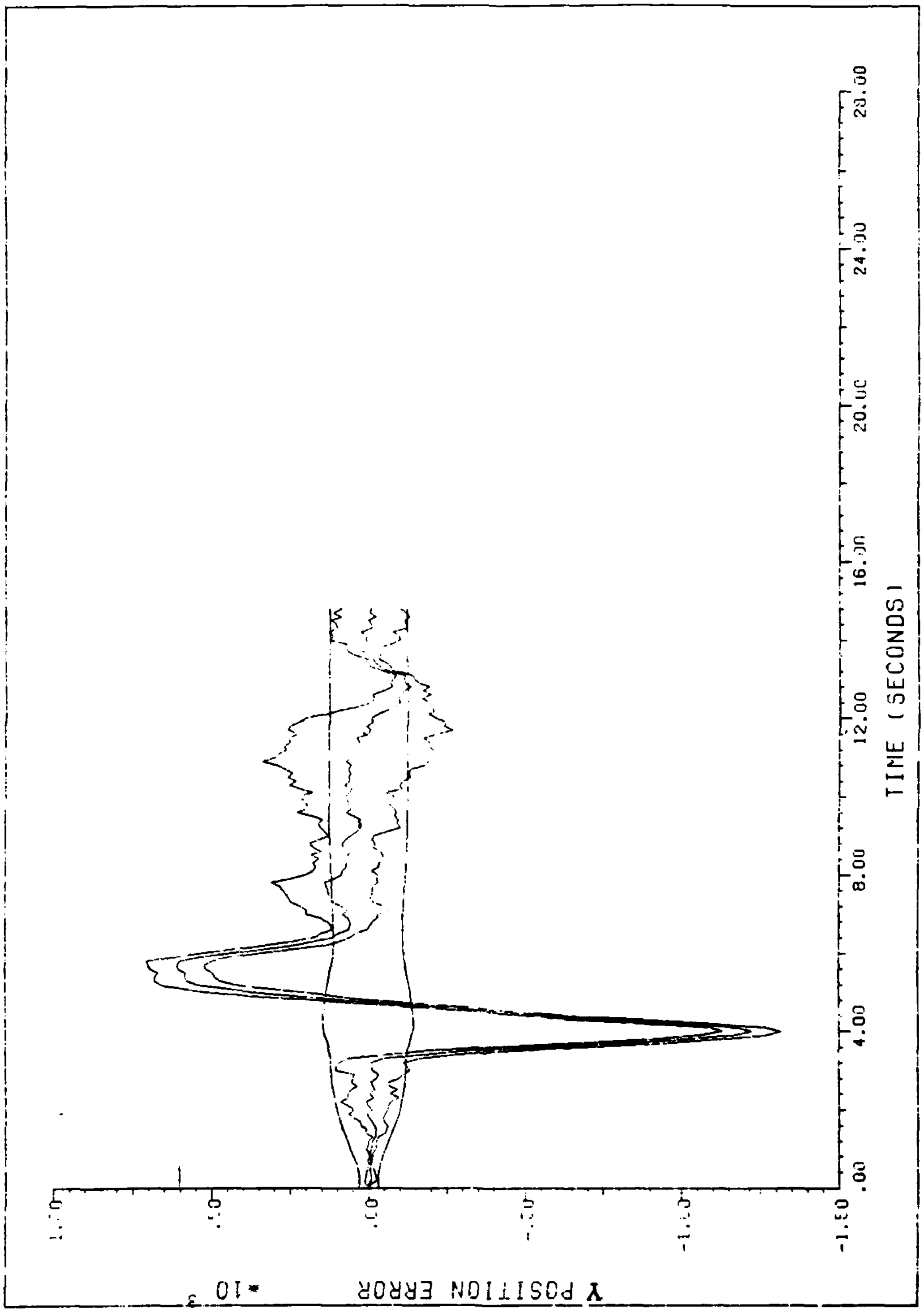


Figure G.7.2.b



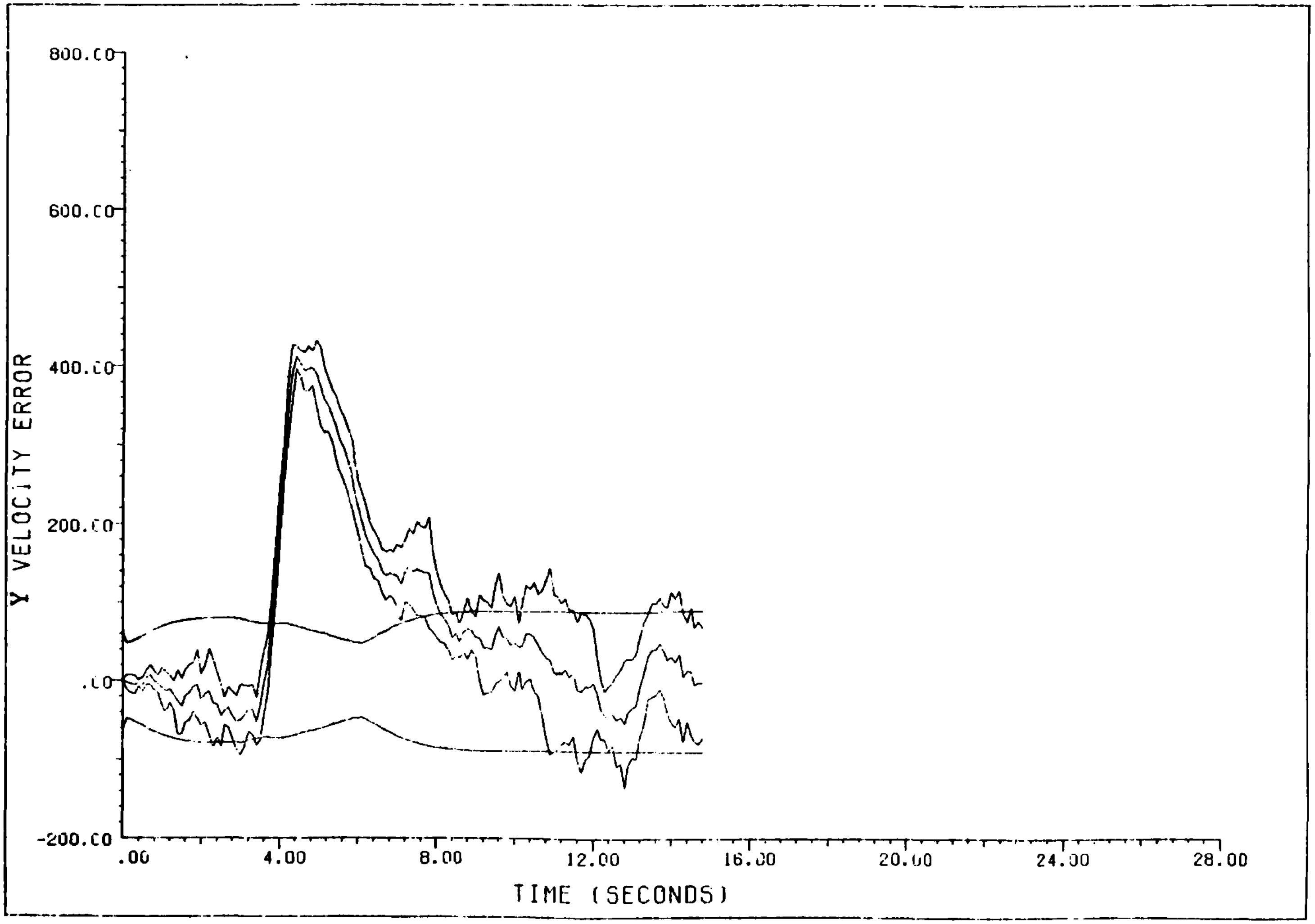
APQ120 Q=VAR TAU=.143 6MEAS 5 RUNS

Figure G.7.2.c



APG12D Q = VAR TAU= .143 6 MEAS. 5 RUNS

Figure G.7.2.d



APQ120 Q= VAR TAU=.143 6 MEAS 5 RUNS

Figure G.7.2.e

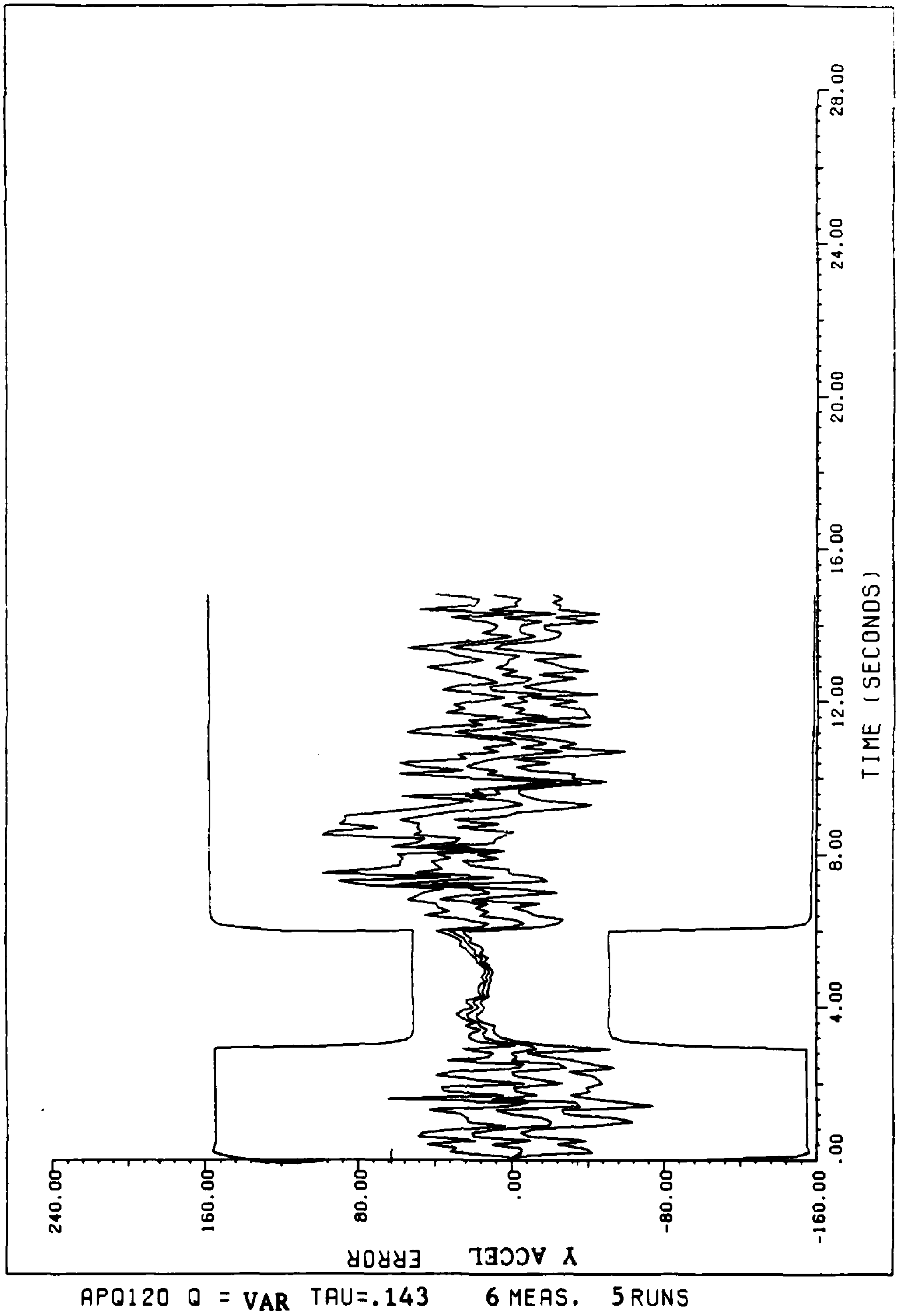

Y ACCEL ERROR
240.00
160.00
80.00
.00
-80.00
-160.00
TIME (SECONDS)
.00
4.00
8.00
12.00
16.00
20.00
24.00
28.00
APQ120 Q = VAR TAU=.143 6 MEAS. 5 RUNS

Figure G.7.2.f

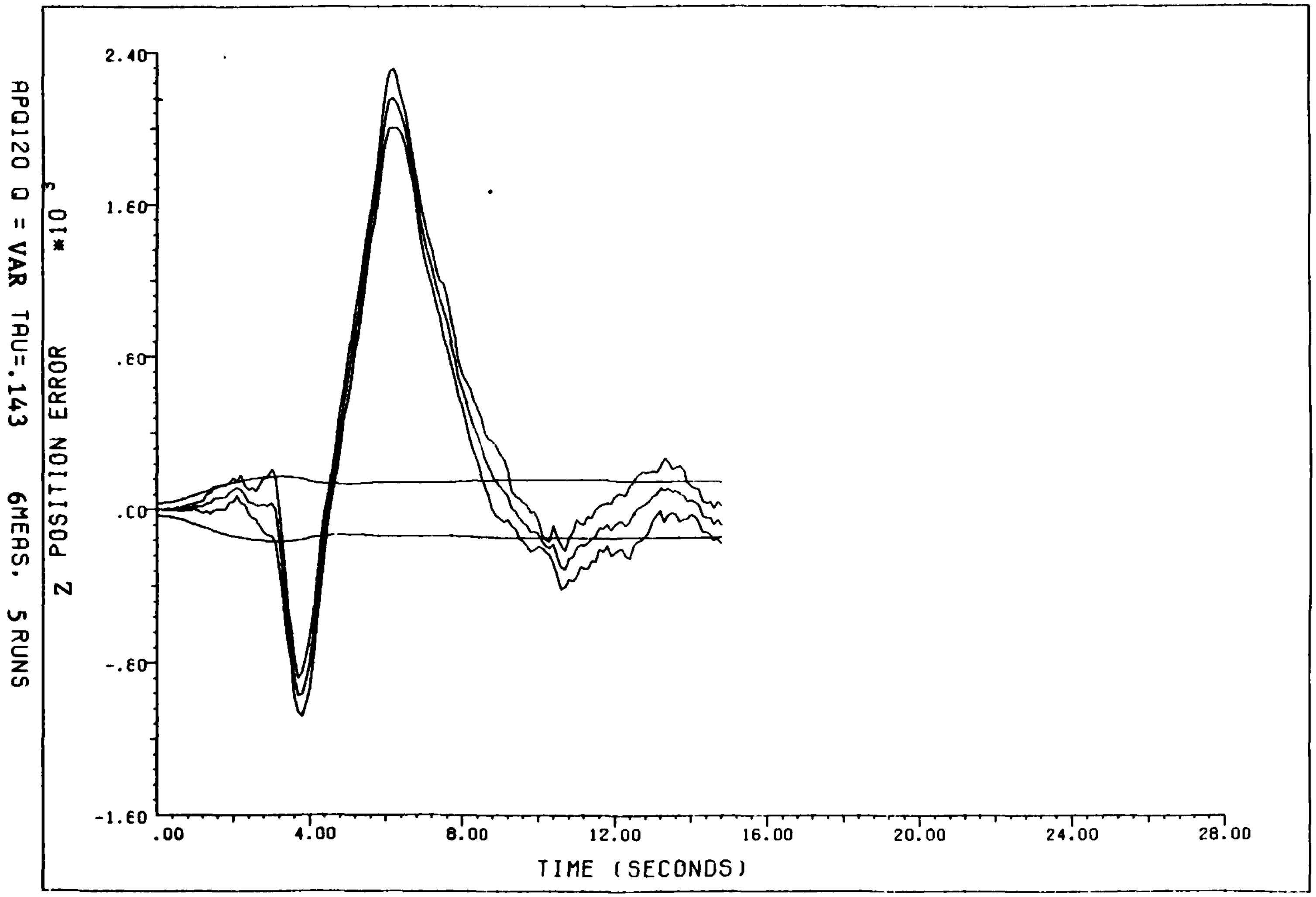
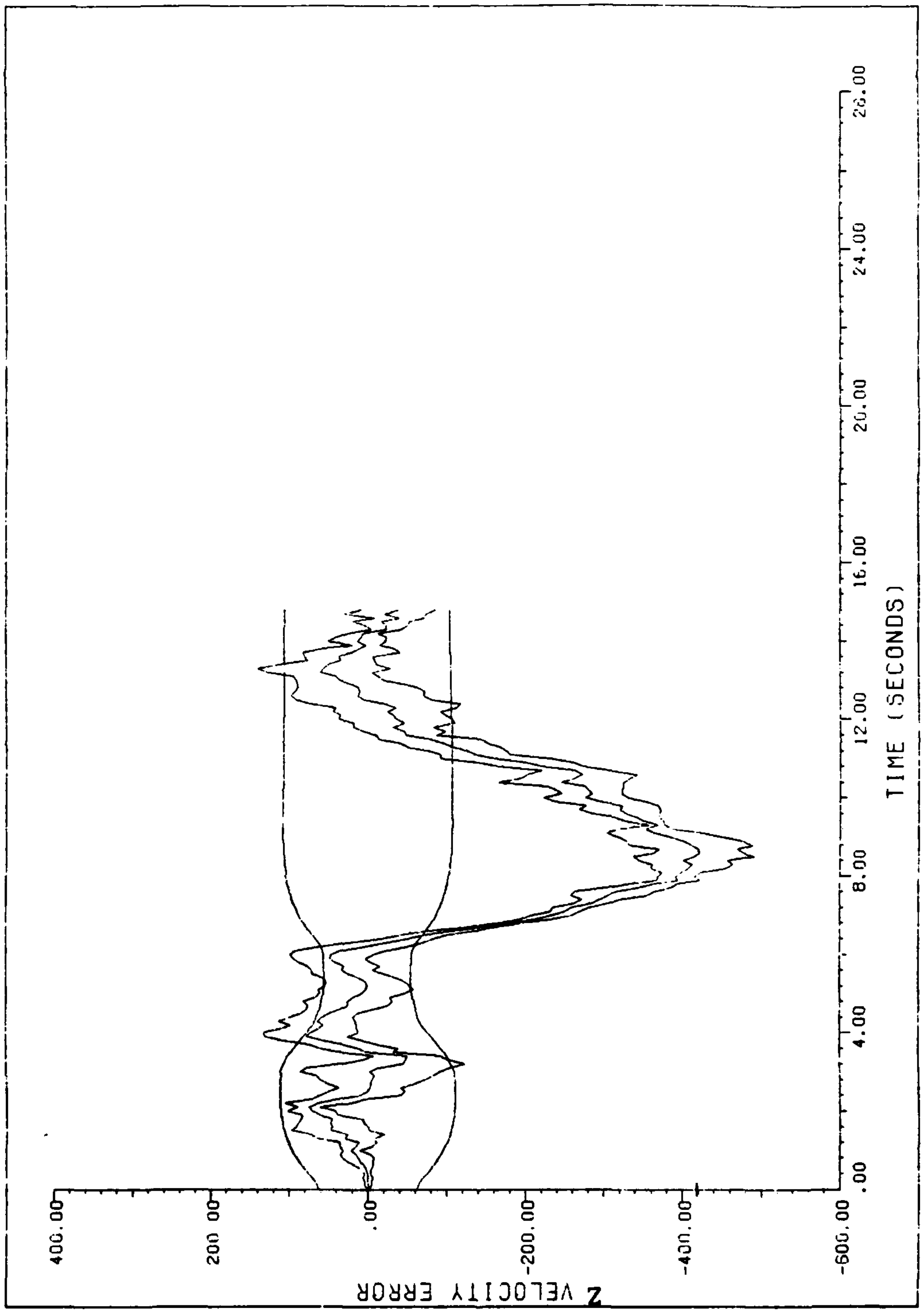

APQ120 Q = VAR TAU=.143 6MEAS. 5 RUNS
Z POSITION ERROR *10 3
2.40
1.60
.80
.00
-.80
-1.60
.00
4.00
8.00
12.00
16.00
20.00
24.00
28.00
TIME (SECONDS)

Figure G.7.2.g



AFQ120 Q=VAR TAU=.143 6 MEAS 5 RUNS

Figure G.7.2.h

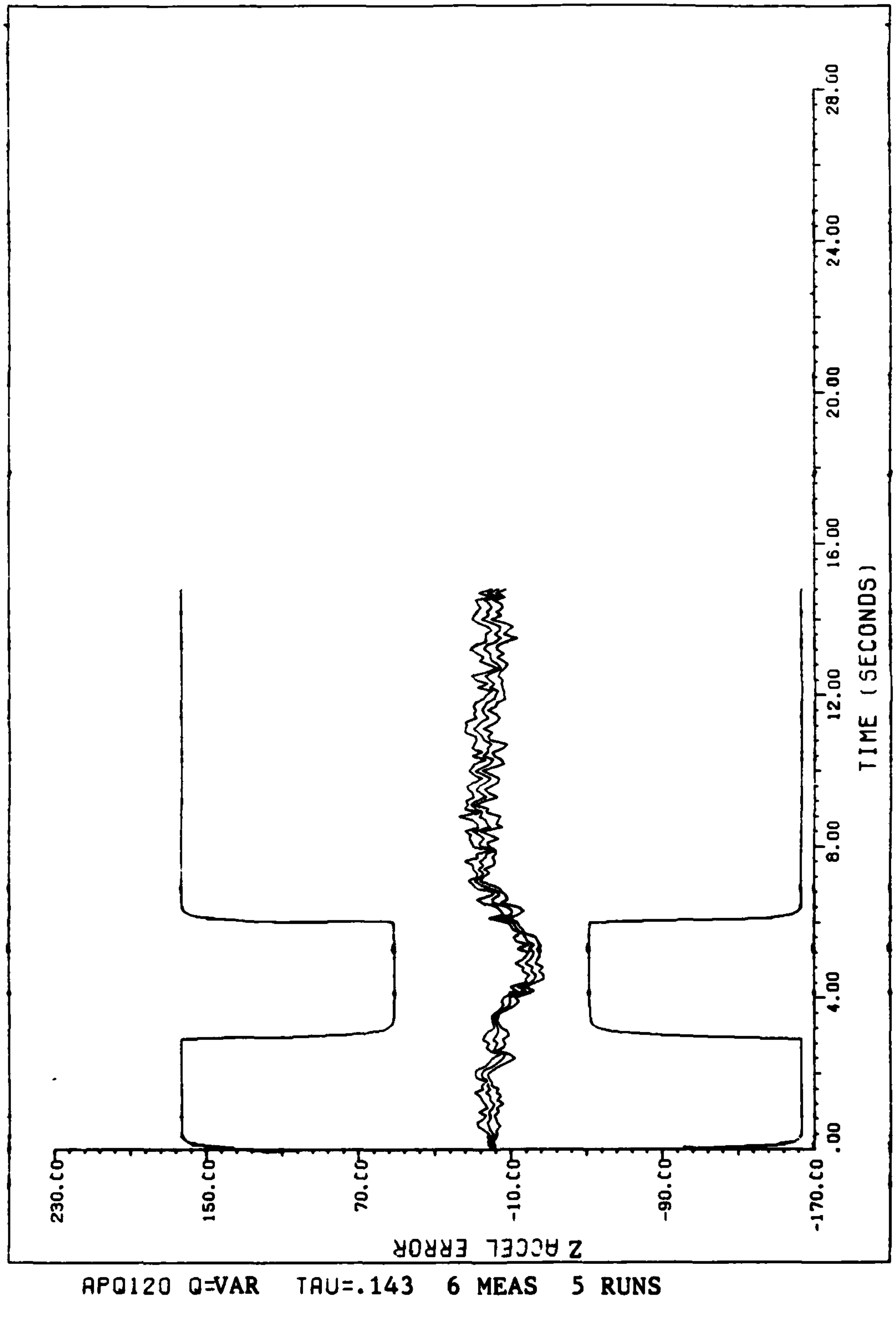

Z ACCEL ERROR
230.00
150.00
70.00
-10.00
-90.00
-170.00
.00
4.00
8.00
12.00
16.00
20.00
24.00
28.00
TIME (SECONDS)
APQ120 Q=VAR TAU=.143 6 MEAS 5 RUNS

Figure G.7.2.1

Figure Set G.7.3 is the same as Figure Set G.2.4

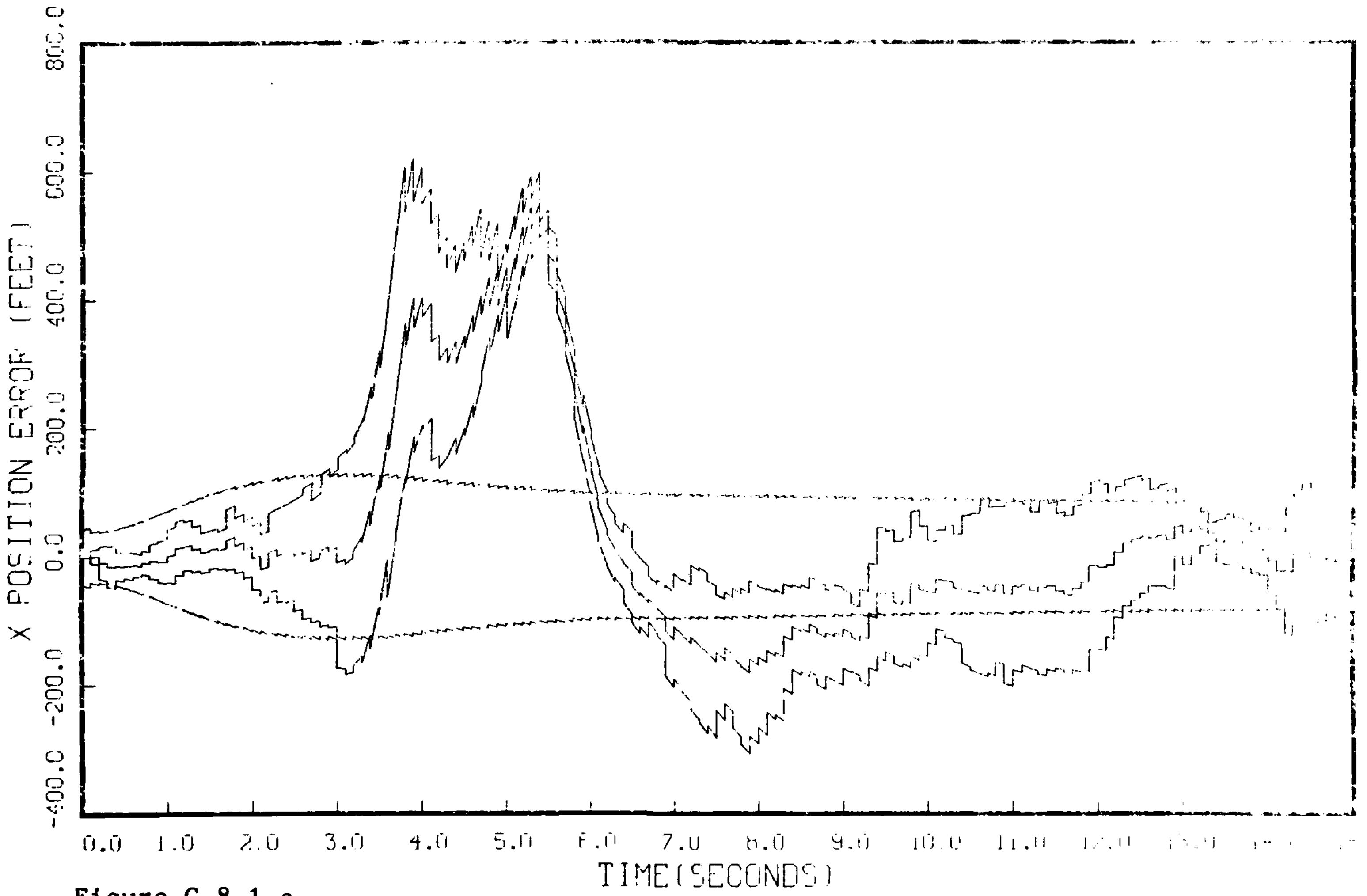


Figure G.8.1.a

STATE 1, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE 10,000.', UPDATE=.1, [illegible]

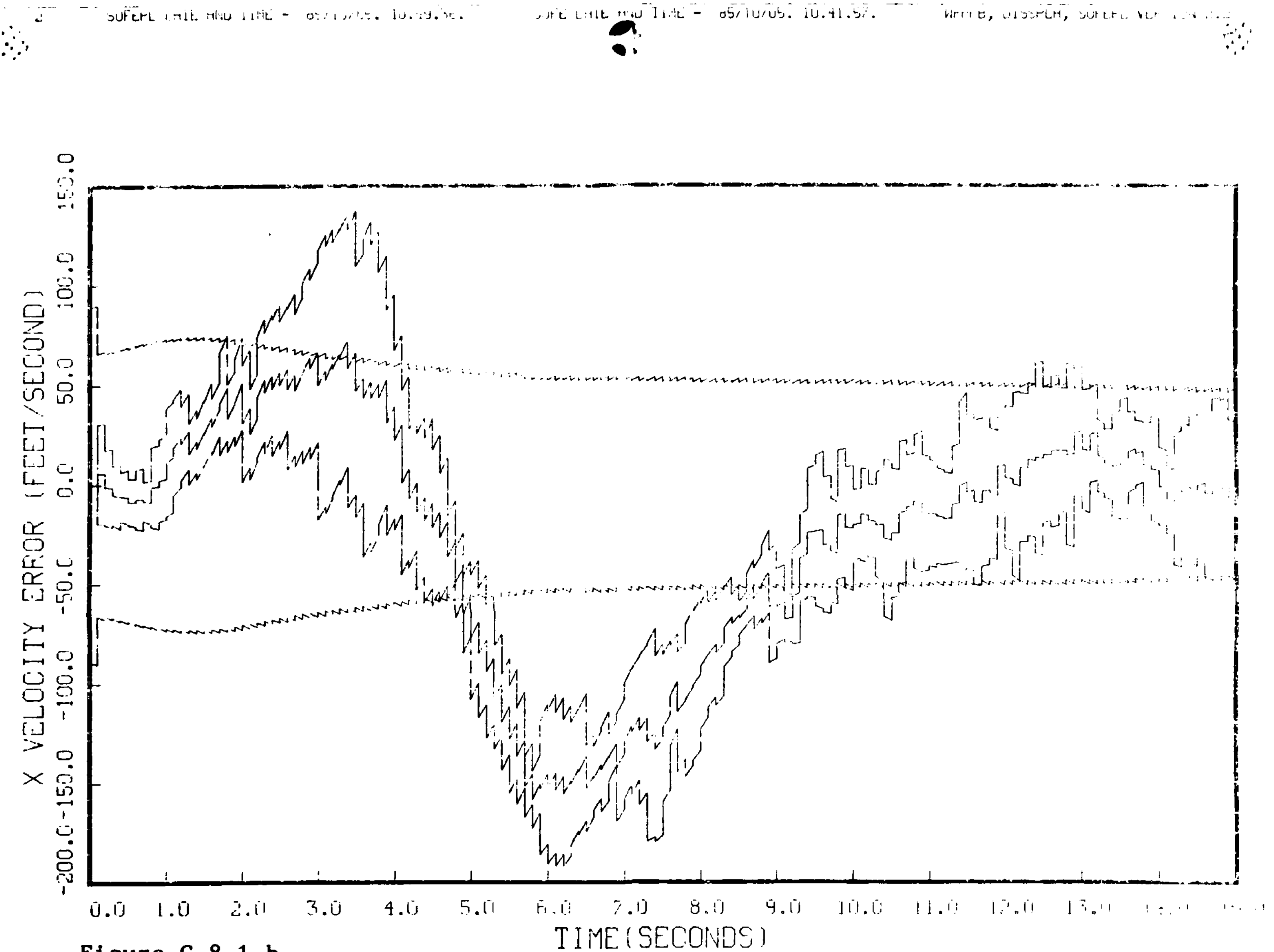


Figure G.8.1.b

STATE 2, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, [illegible] RUNS, NO RADAR [illegible]

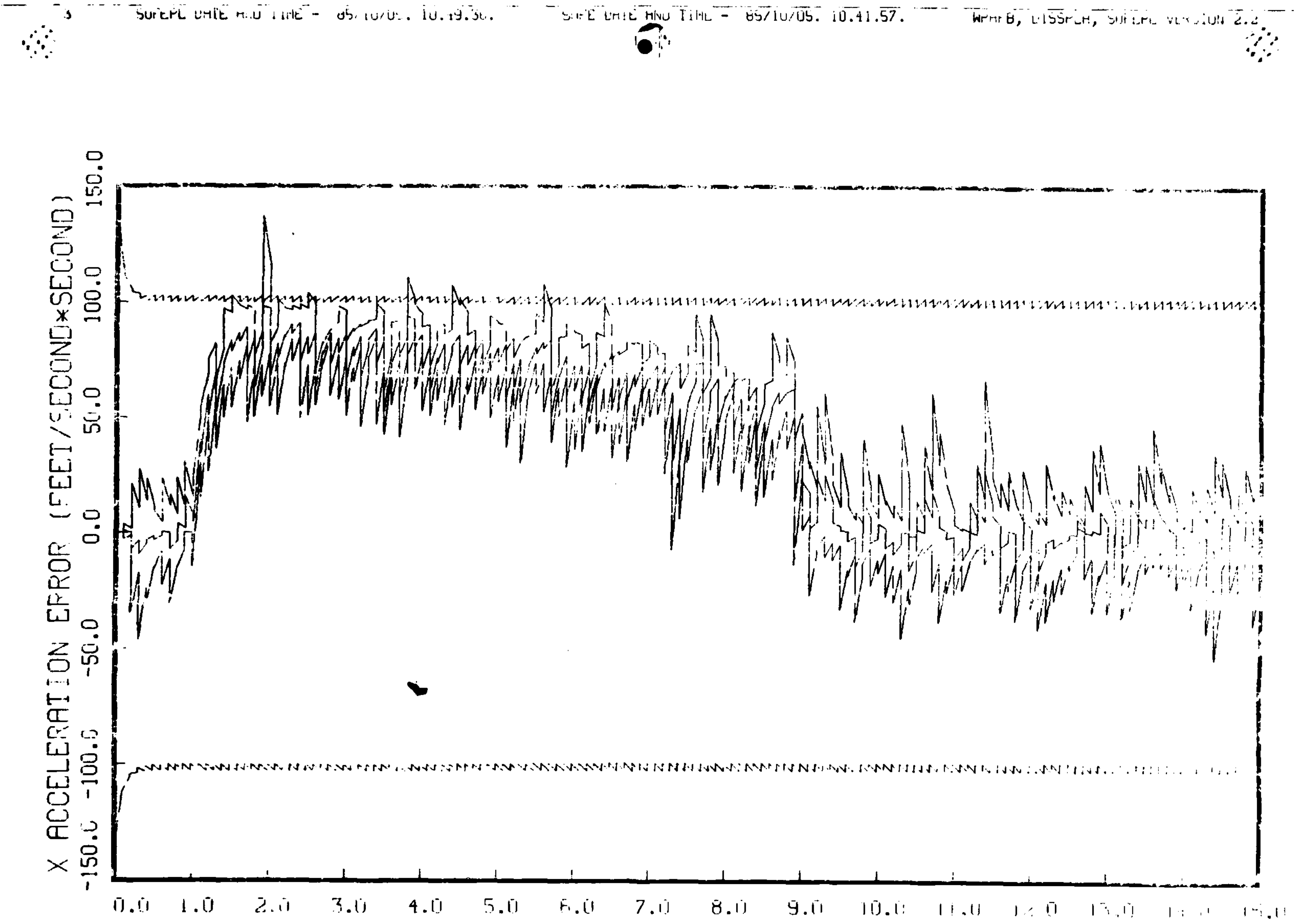


Figure G.8.1.c

STATE 3, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS, NO RADAR LOS

4 SOFEPL DATE AND TIME - 85/10/05. 10.49.36. SOFE DATE AND TIME - 85/10/05. 10.41.57. WPAFB, DISSPLA, SOFEPL VERSION 2.4

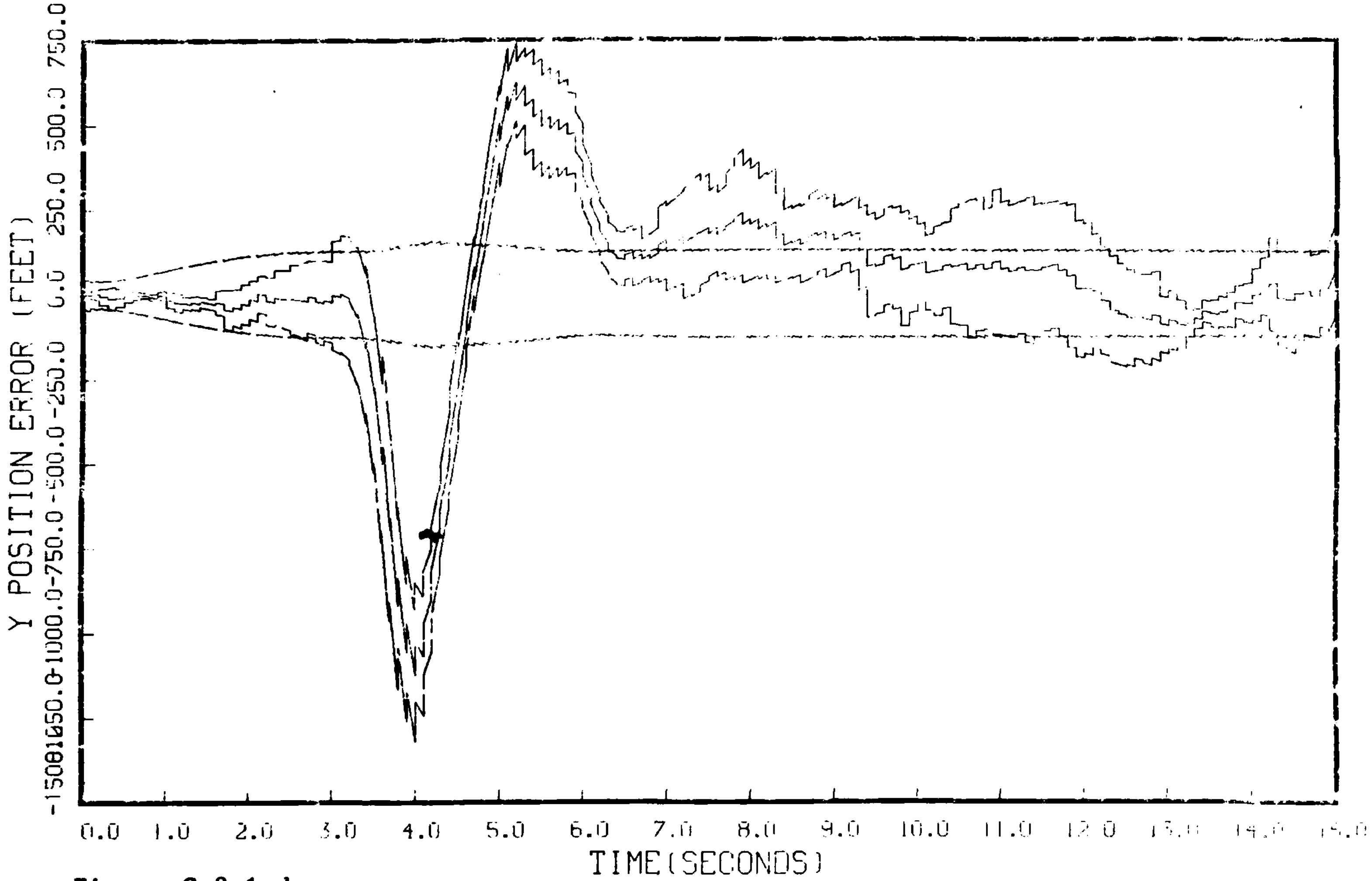


Figure G.8.1.d
STATE 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APG 120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS, NO RADAR LAG

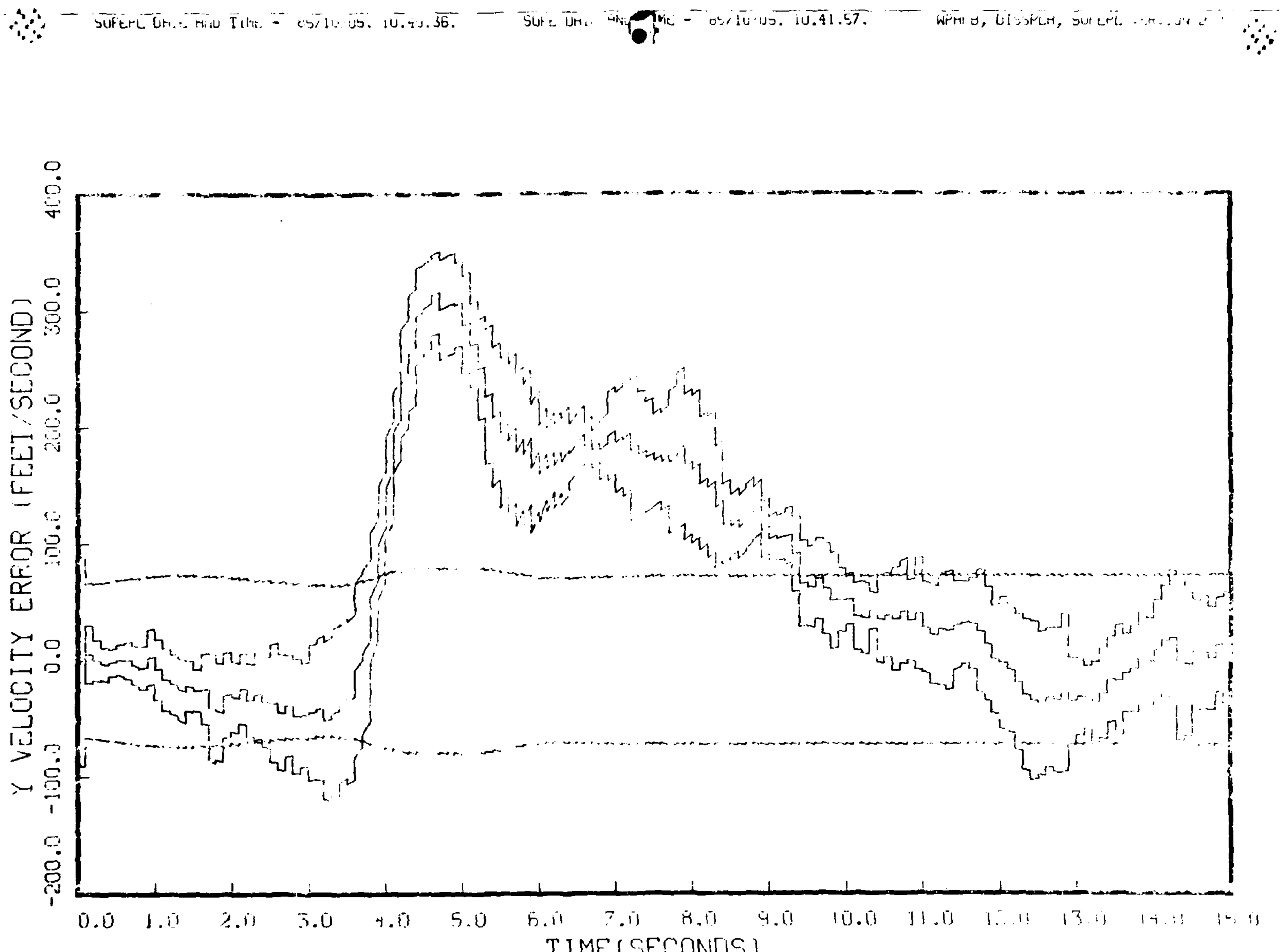


Figure G.8.1.e

STATE 5, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE 10,000.', UPDATE=.1, 5 RUNS, NO RADAR LAG

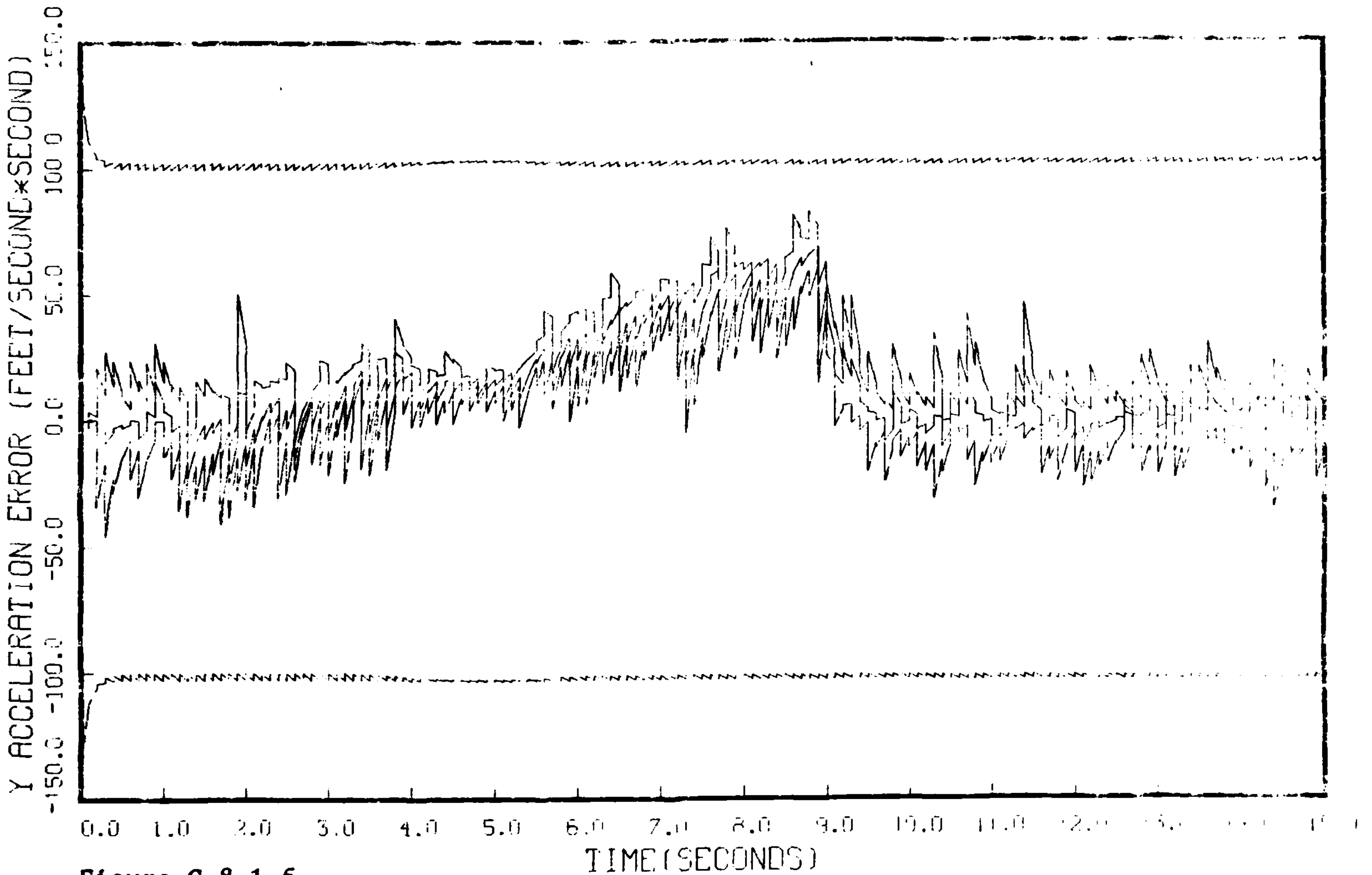


Figure G.8.1.f

STATE 6, Q(1)-Q(2)=Q(3)-149300., TAU(1)=.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE 10,000.', UPDATE-.1, [illegible]

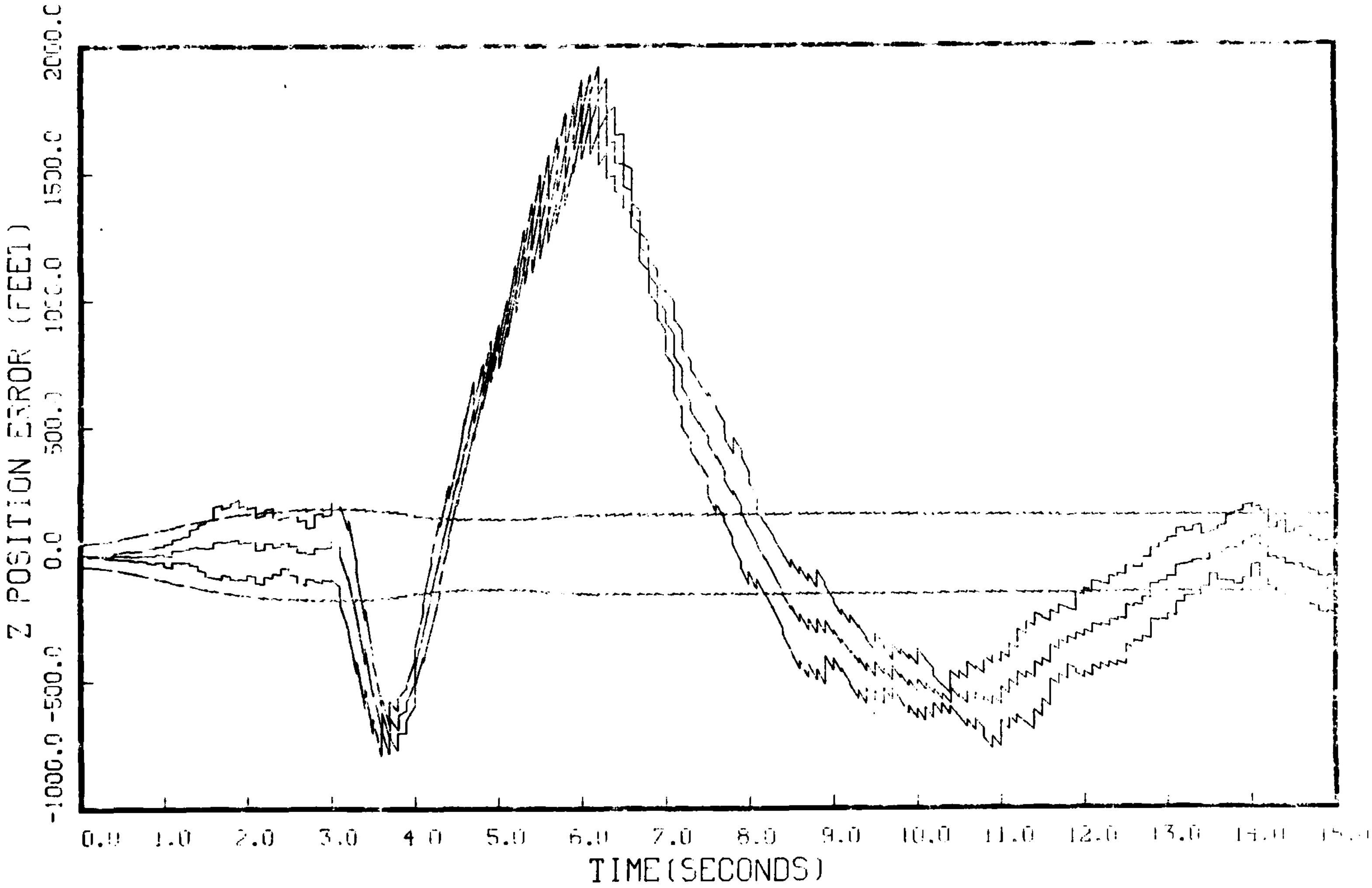


Figure G.8.1.g
STATE 7, Q(1)=Q(2)=Q(3)-149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE=.1, 5 RUNS, NO RADAR LAG

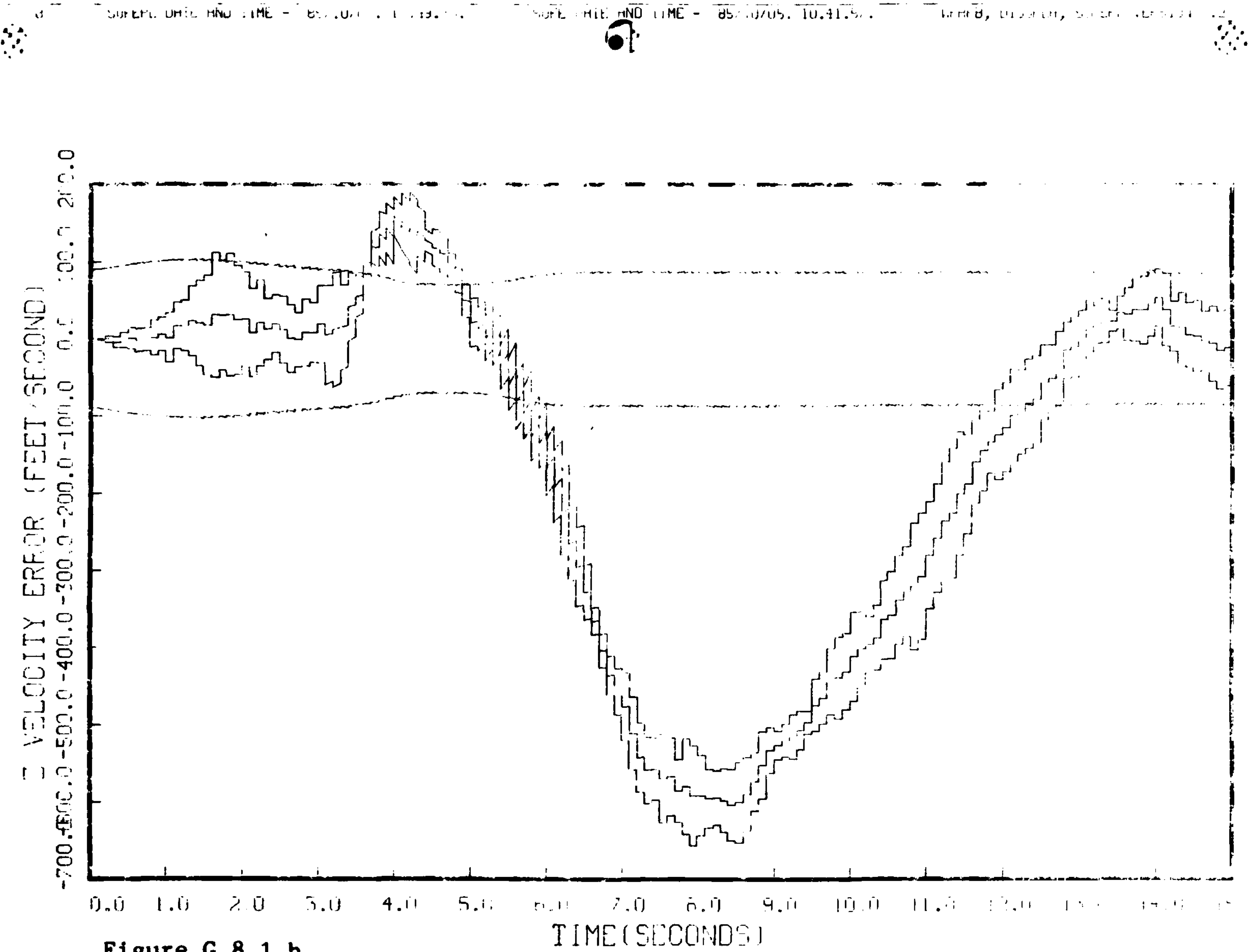


Figure G.8.1.h

STATE 8, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL RUNS
APG 120, BEAM ATTACK, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS, [illegible]

9 SUPERB DATE AND TIME - [illegible] 11.49.[illegible] SUPE DATE AND TIME - 85/10/05. 10.41.57. WPAFB, DISPLAY, SUPERB VER [illegible]

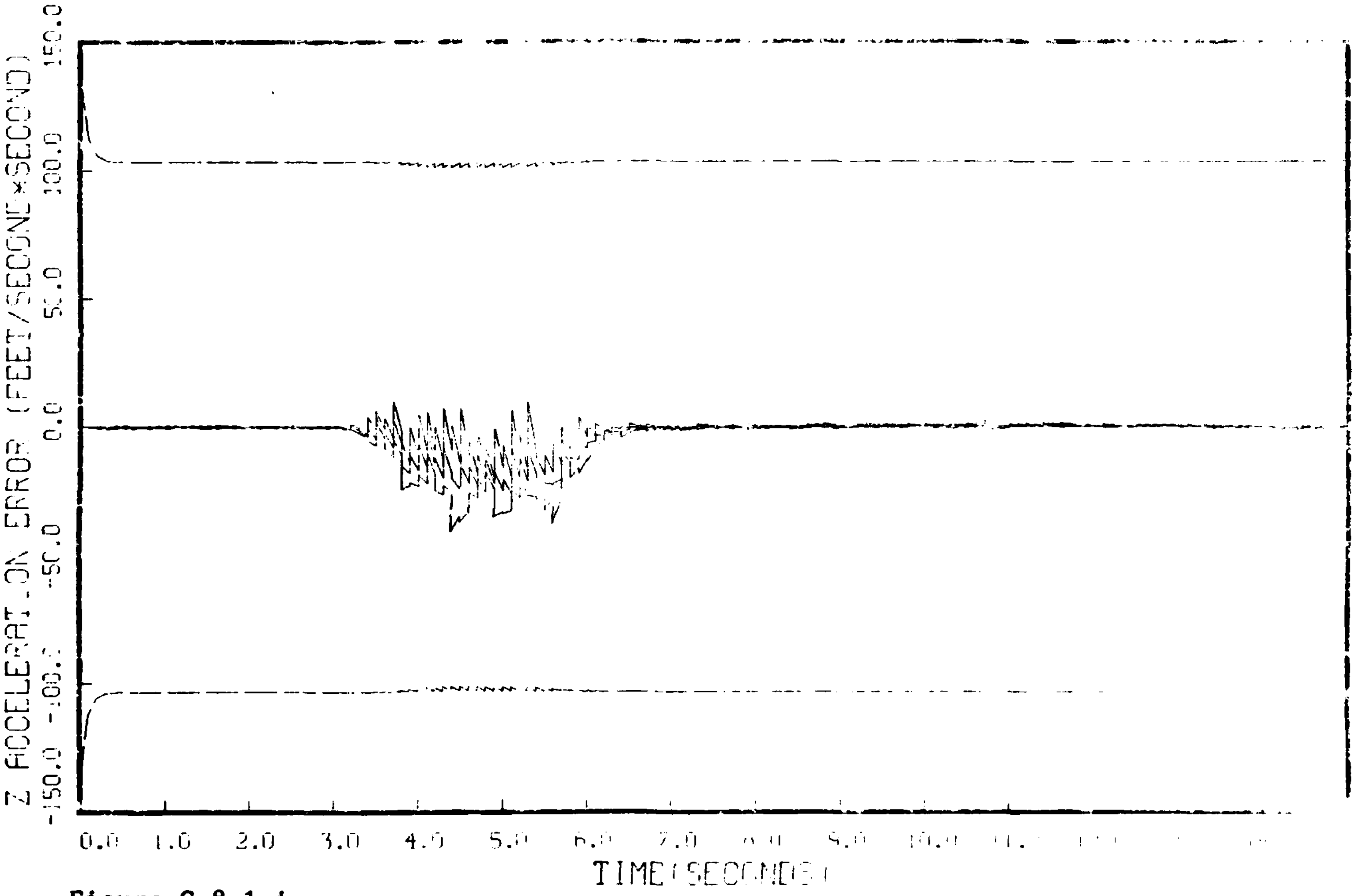


Figure G.8.1.i
STATE 9, Q(1)-Q(2)-Q(3)-149500., TAU(1)-.143,TAU(2-3)-.14[illegible], ALL ML[illegible]
APC 120, BEAM ATTACK, INITIAL RANGE 40,000. , UPDATE-.1, [illegible]

78 PS9 1 SOFEPL DATE AND TIME - 85/10/05. 01.34.46. SOFE DATE AND TIME - 85/10/04. 23.10.18. WPAFB, DISSPLA, SOFEPL VERSION

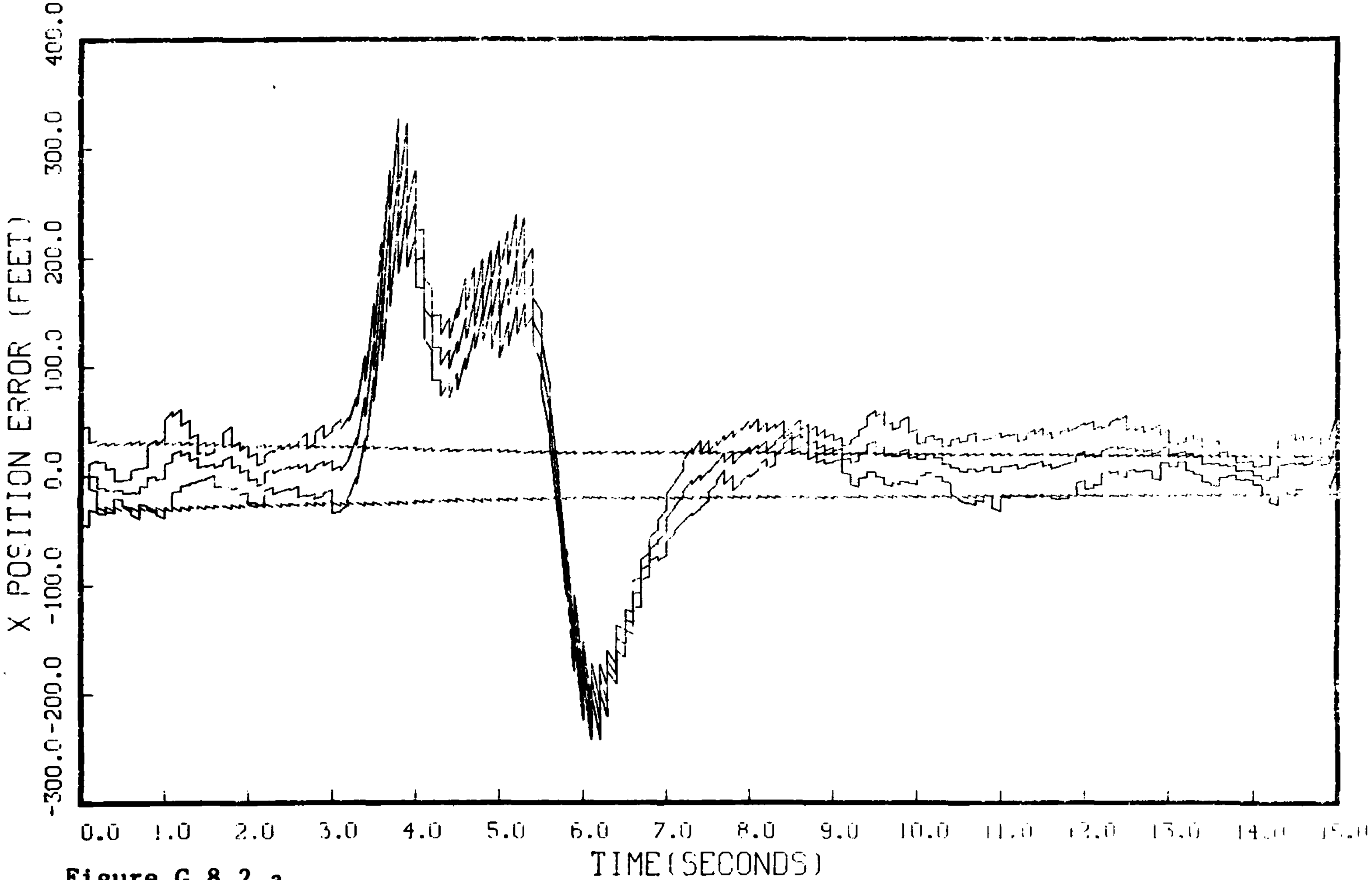


Figure G.8.2.a

STATE 1, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS, NO RADAR LAG

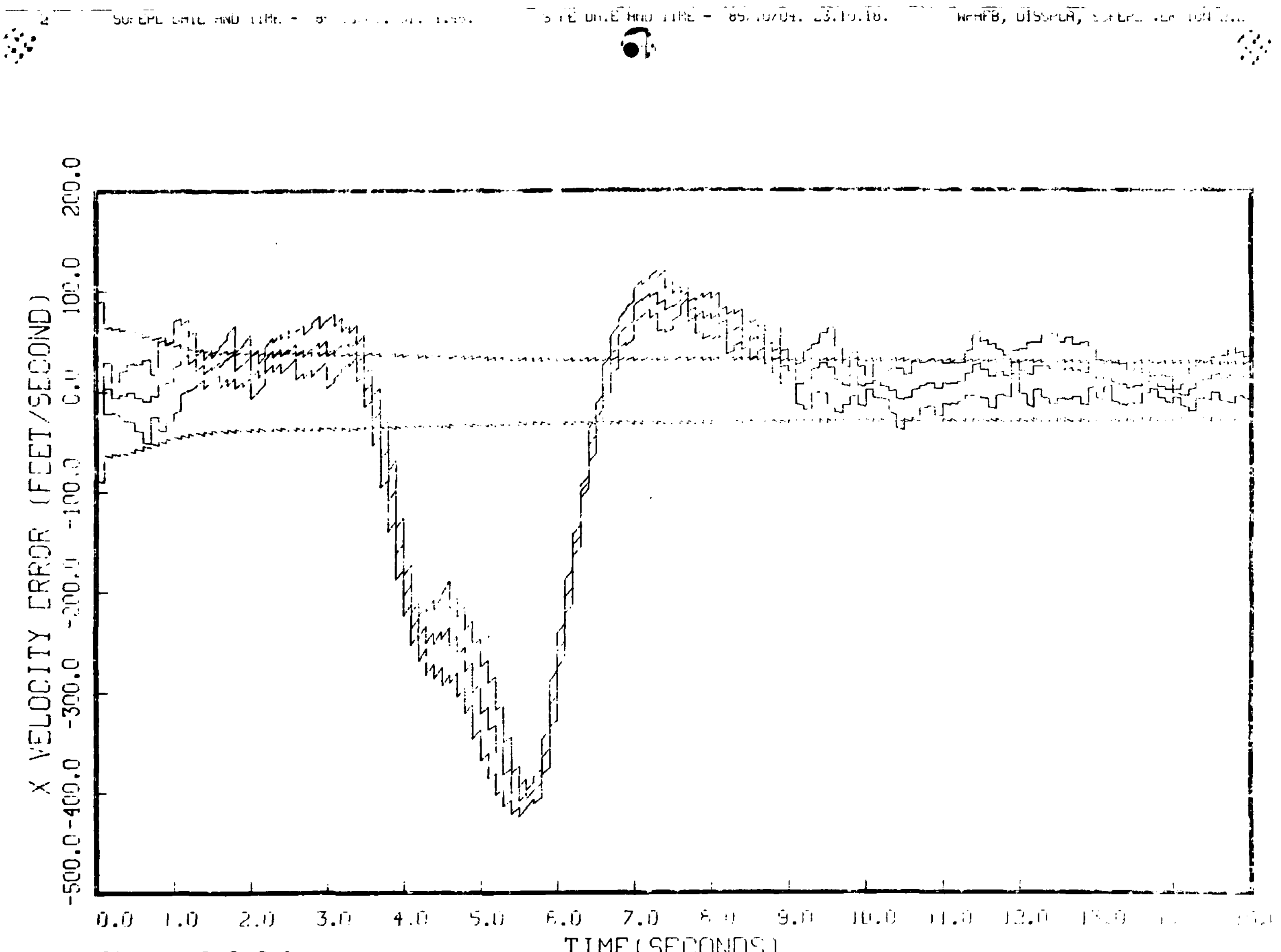


Figure G.8.2.b

STATE 2, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS, NO [illegible]

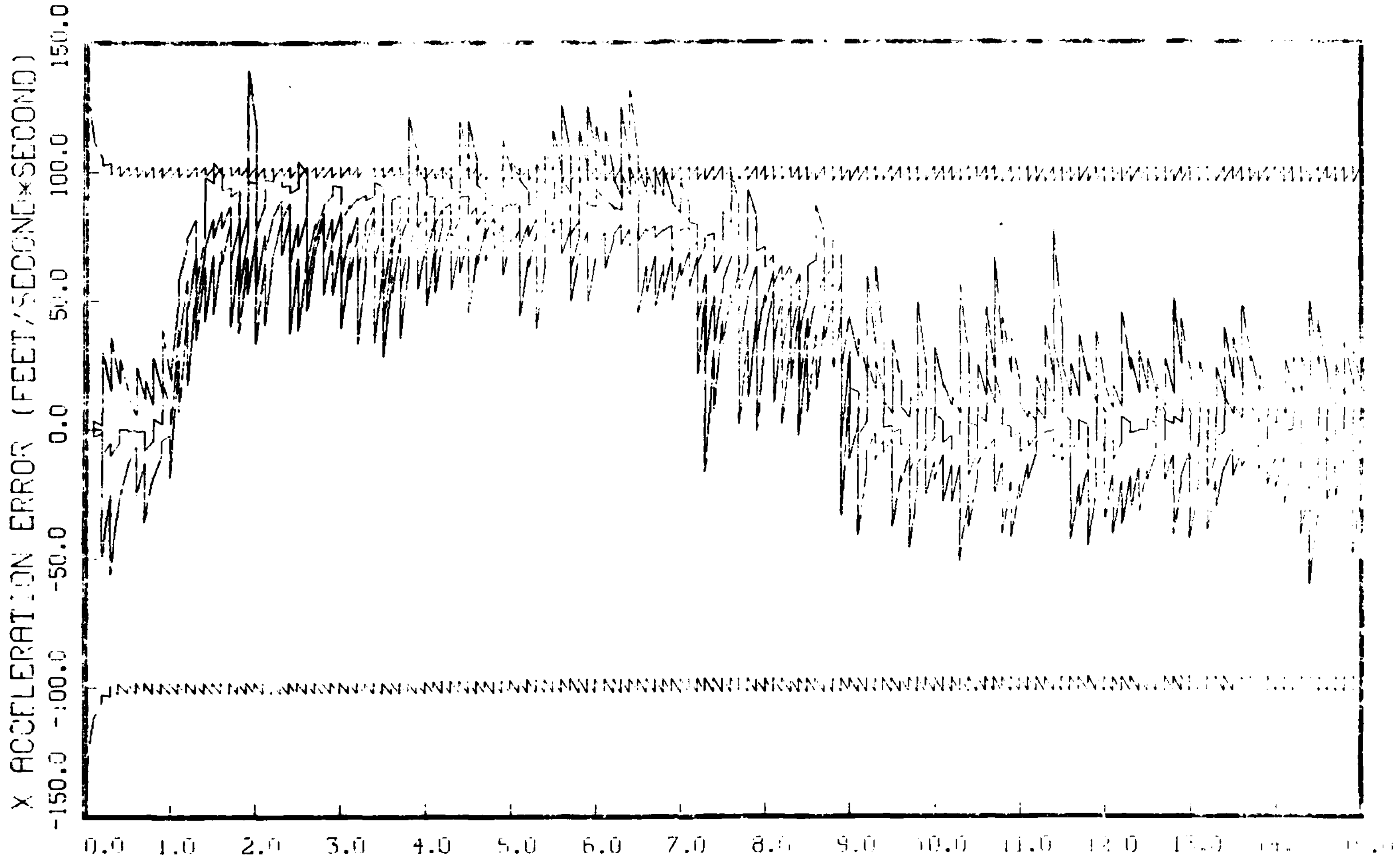


Figure G.8.2.c

STATE 3, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS, NO RADAR [illegible]

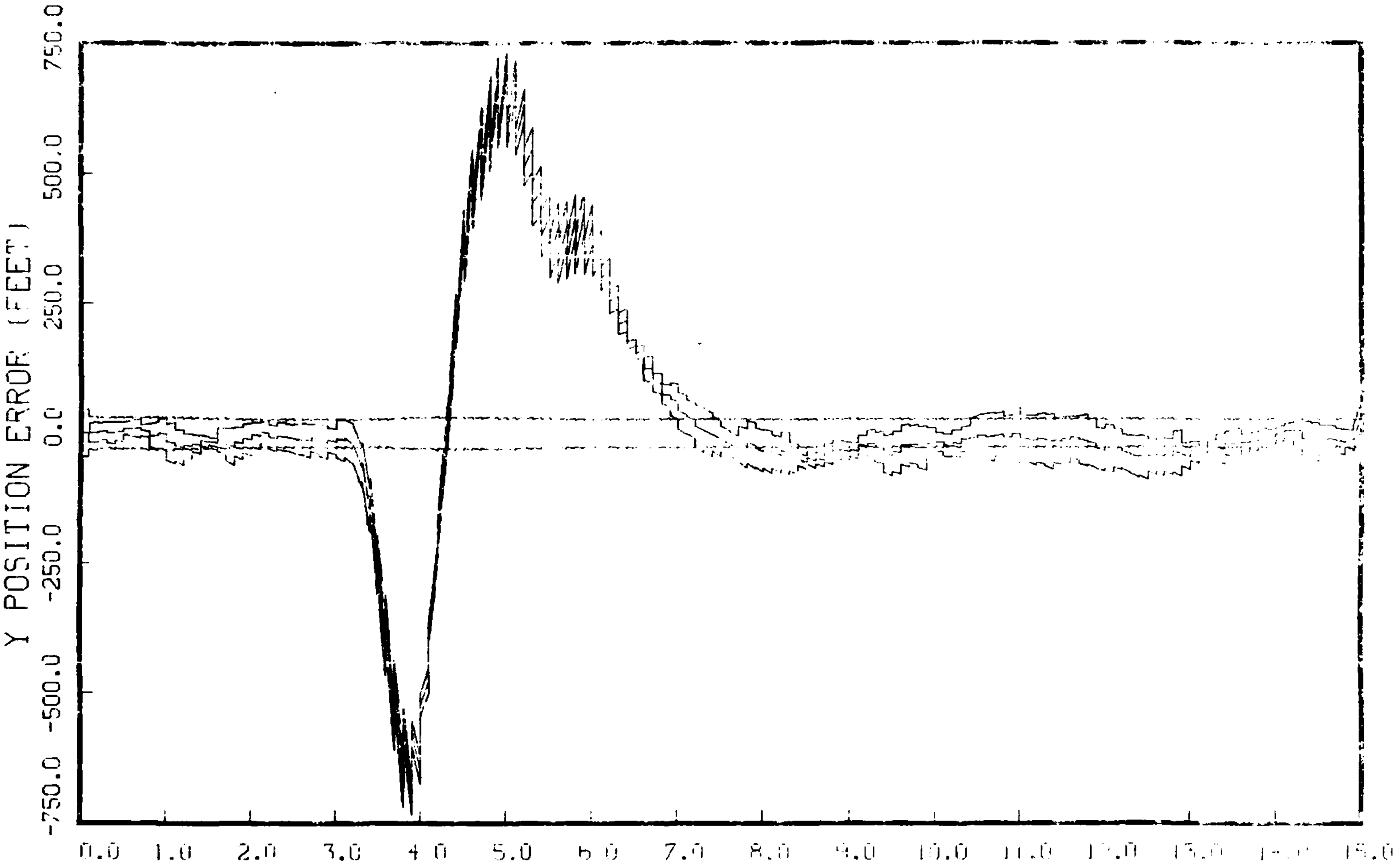


Figure G.8.2.d

STATE 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APC 120, BEAM ATTACK, INITIAL RANGE 40,000.', UPDATE-.1, 5 RUNS, NO RADAR [illegible]

5 SUPER DATE AND TIME - 85/1. 05. 1.:.:46. SUB. DATE AND TIME - 85/10 04. 23.10..3. WPHM, DISSPLH, SUBPL [illegible]

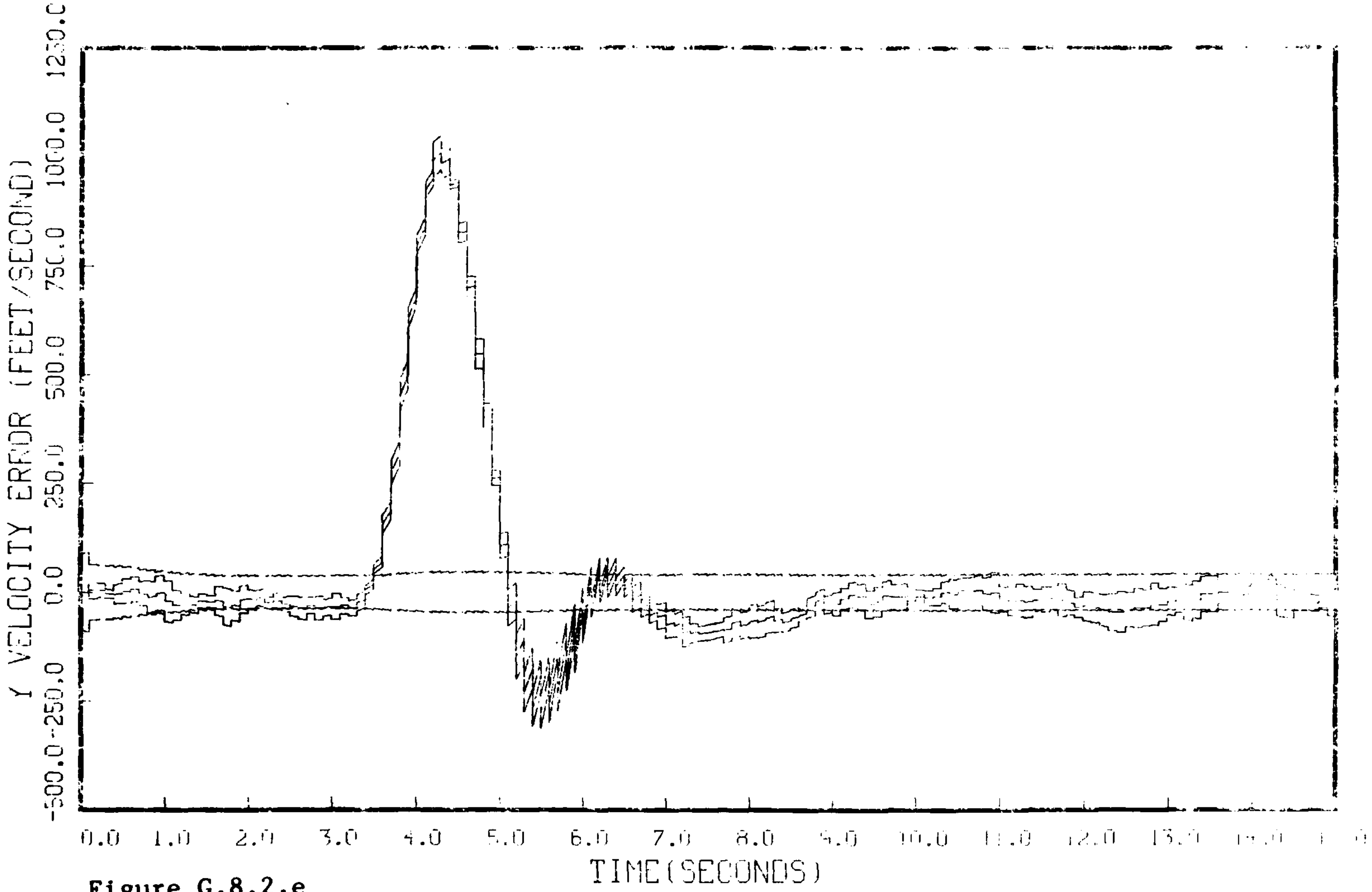


Figure G.8.2.e

STATE 5, Q(1)=Q(2)=Q(3)=140300., TAU(1)=.143,TAU(2-3)=.143. ALL MEAS
AFO 120, DEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-.1, 5 RUNS, NO RADAR [illegible]

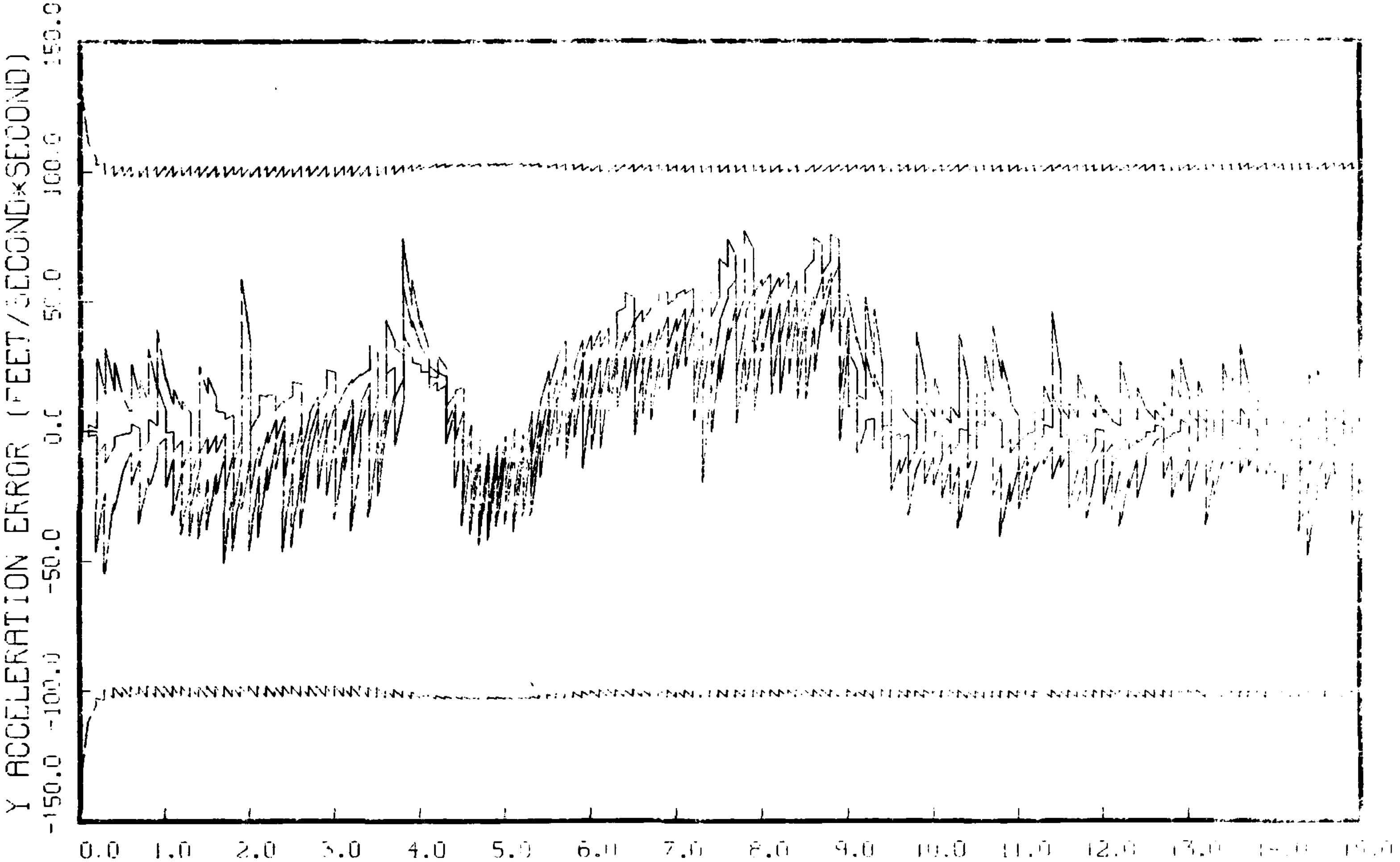


Figure G.8.2.f

STATE 6, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
AFO 120, BEAM ATTACK, INITIAL RANGE=10,000.', UPDATE=.1, 5 RUNS, NO RADAR

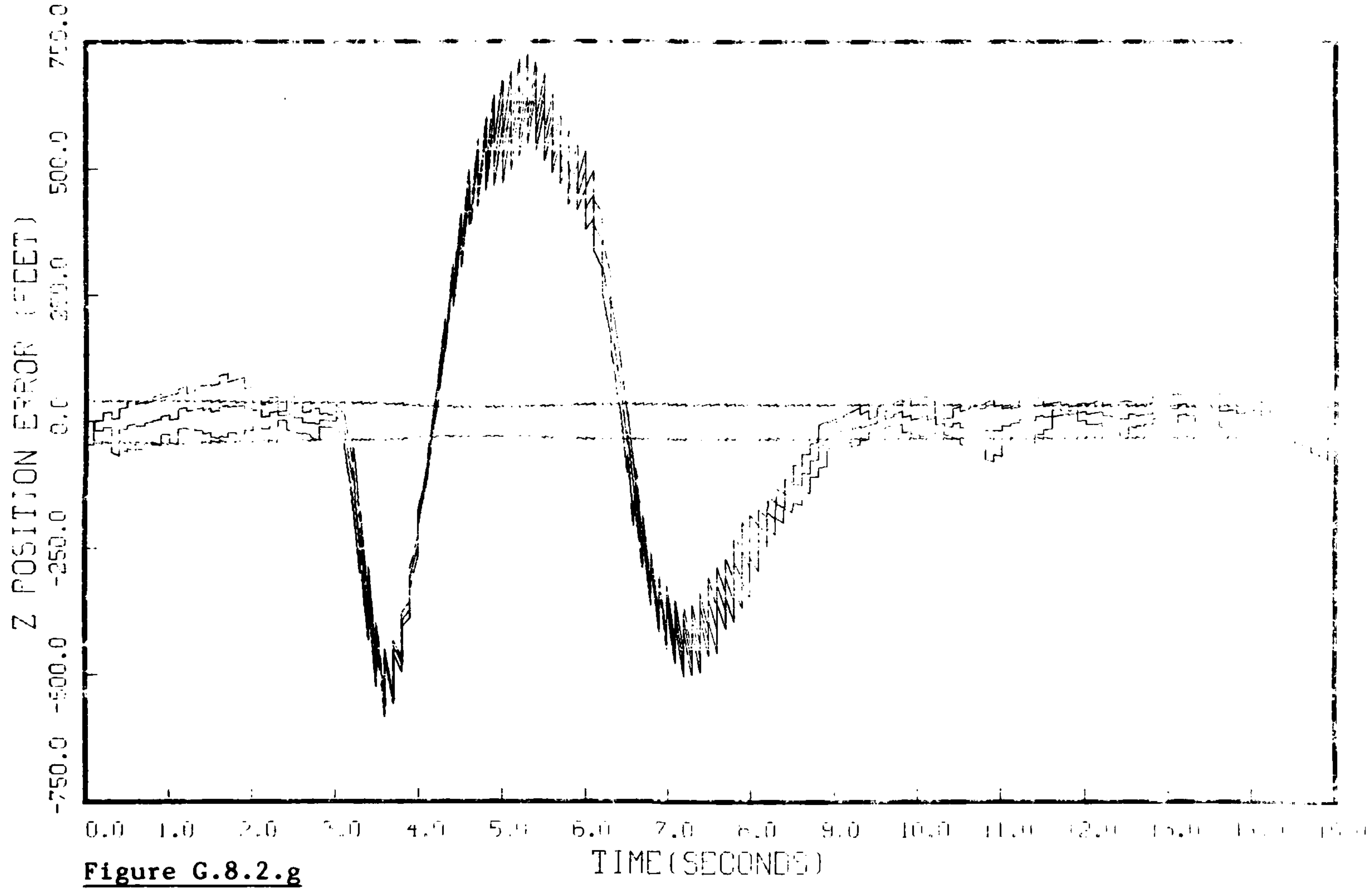


Figure G.8.2.g

STATE 7, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM AIMING, INITIAL RANGE=10,000.', UPDATE=.1, 5 [illegible], NO RADAR [illegible]

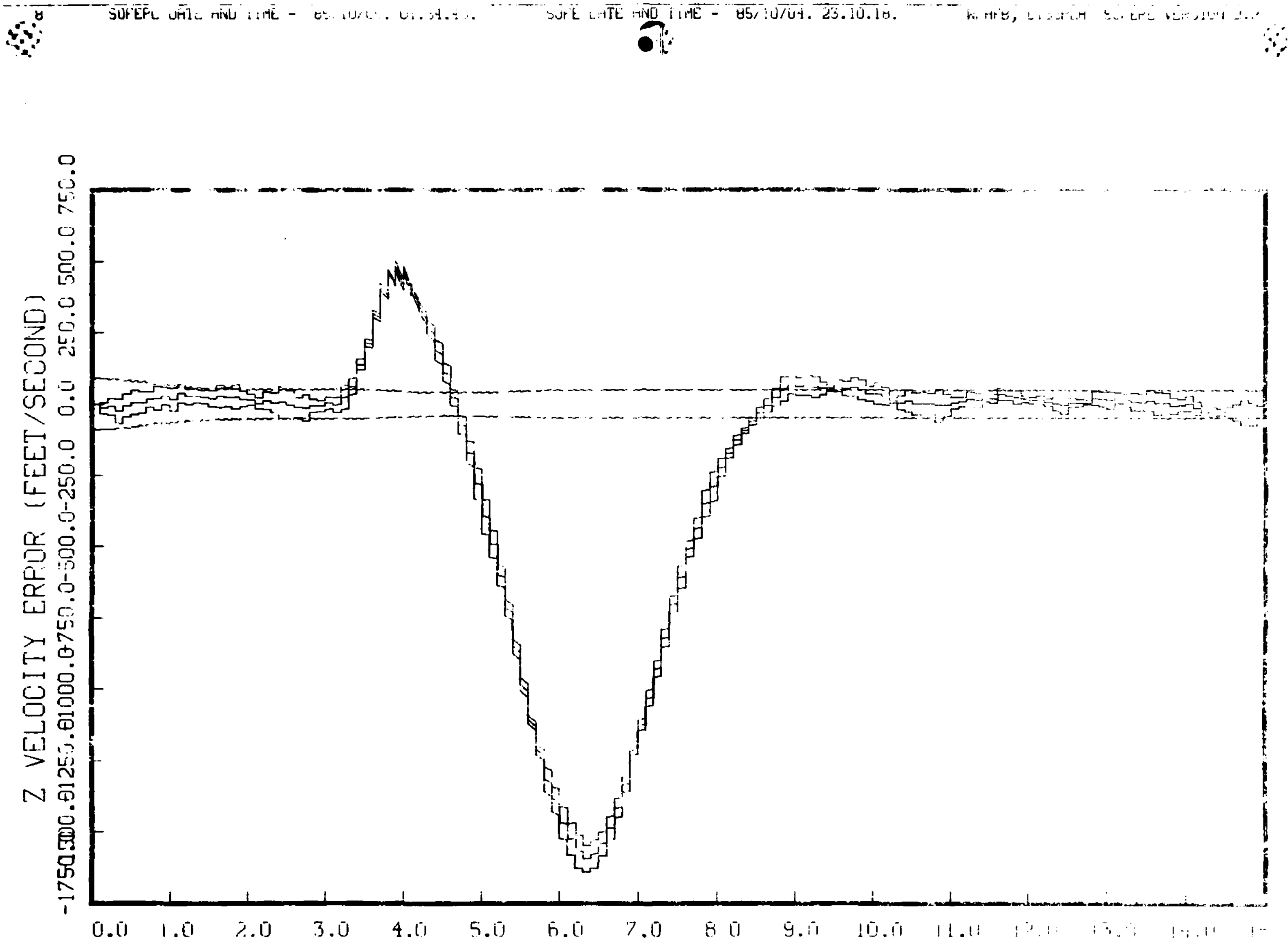


Figure G.8.2.h

STATE 8, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS, NO [illegible]

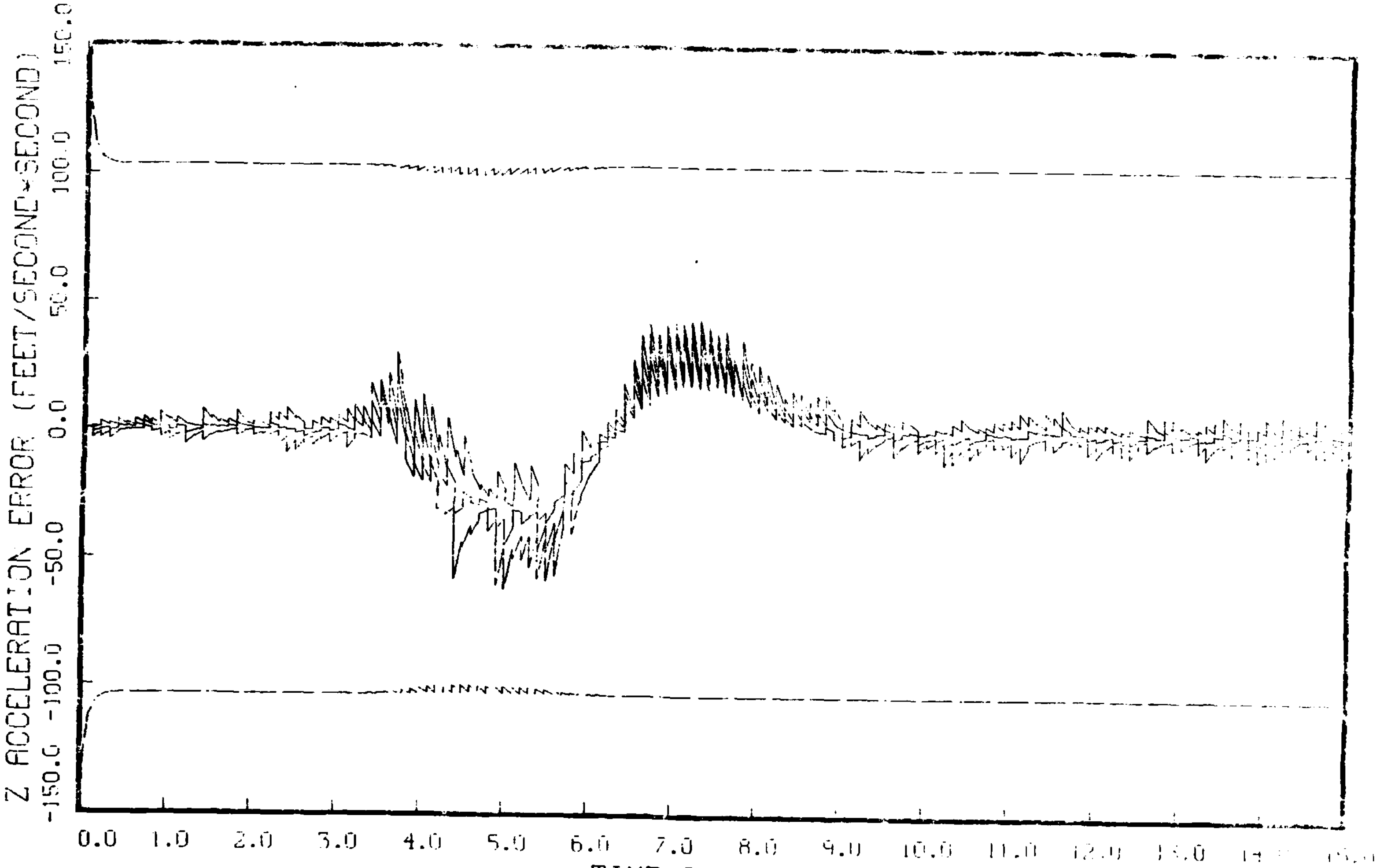


Figure G.8.2.i

STATE 9, Q(1)=Q(2)=Q(3)=149300., TAU(1)=.143,TAU(2-3)=.143, ALL MEAS
APC 120, BEAM ATTACK, INITIAL RANGE=40,000.', UPDATE=.1, 5 RUNS, NO RADAR [illegible]

Figure Set G.8.3 is the same as Figure Set G.3.2

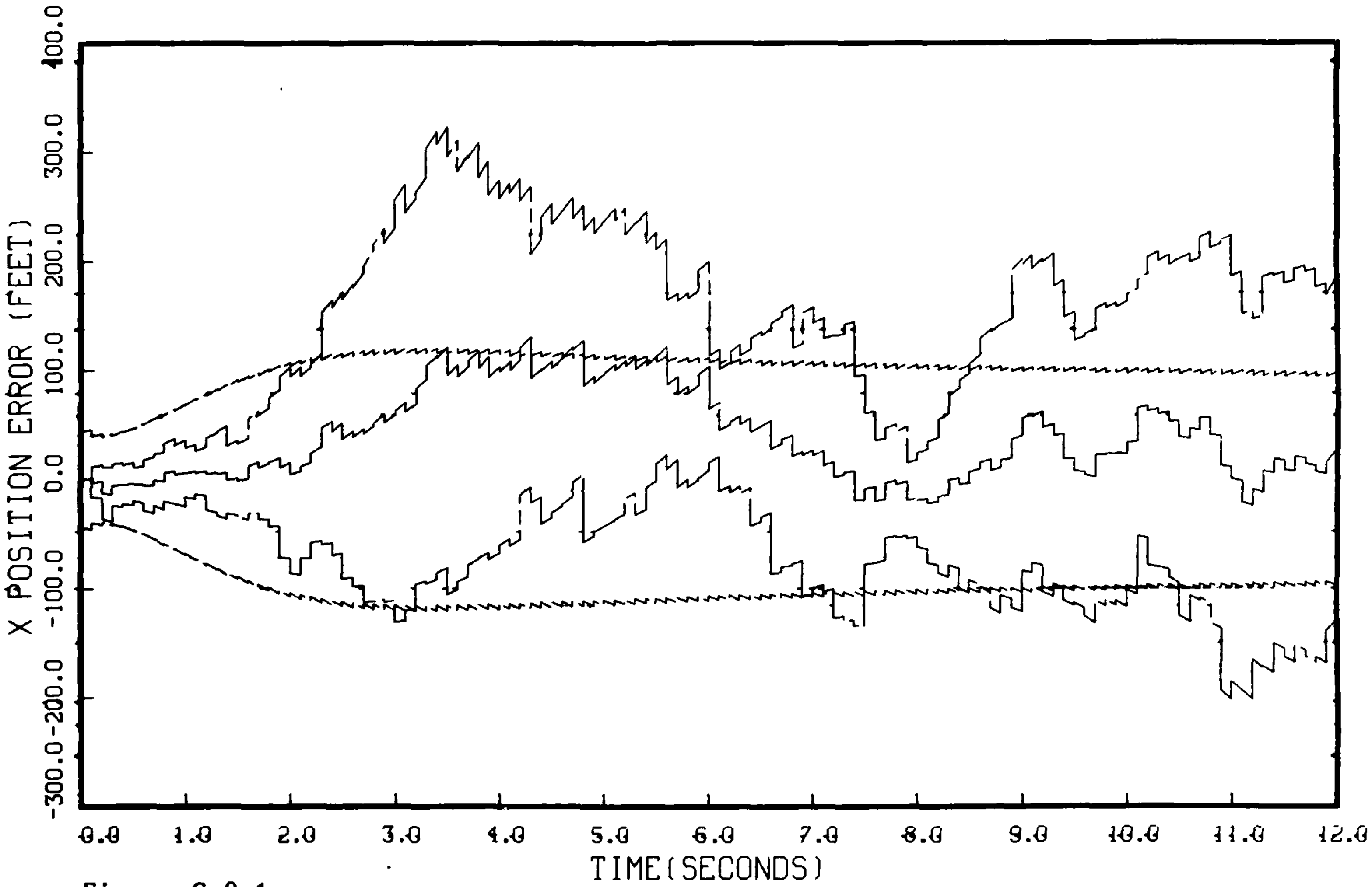


Figure G.9.1.a
STATE 1, O(1)-O(2)-O(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.1, 5 RUN, NO FIGHTER MANS

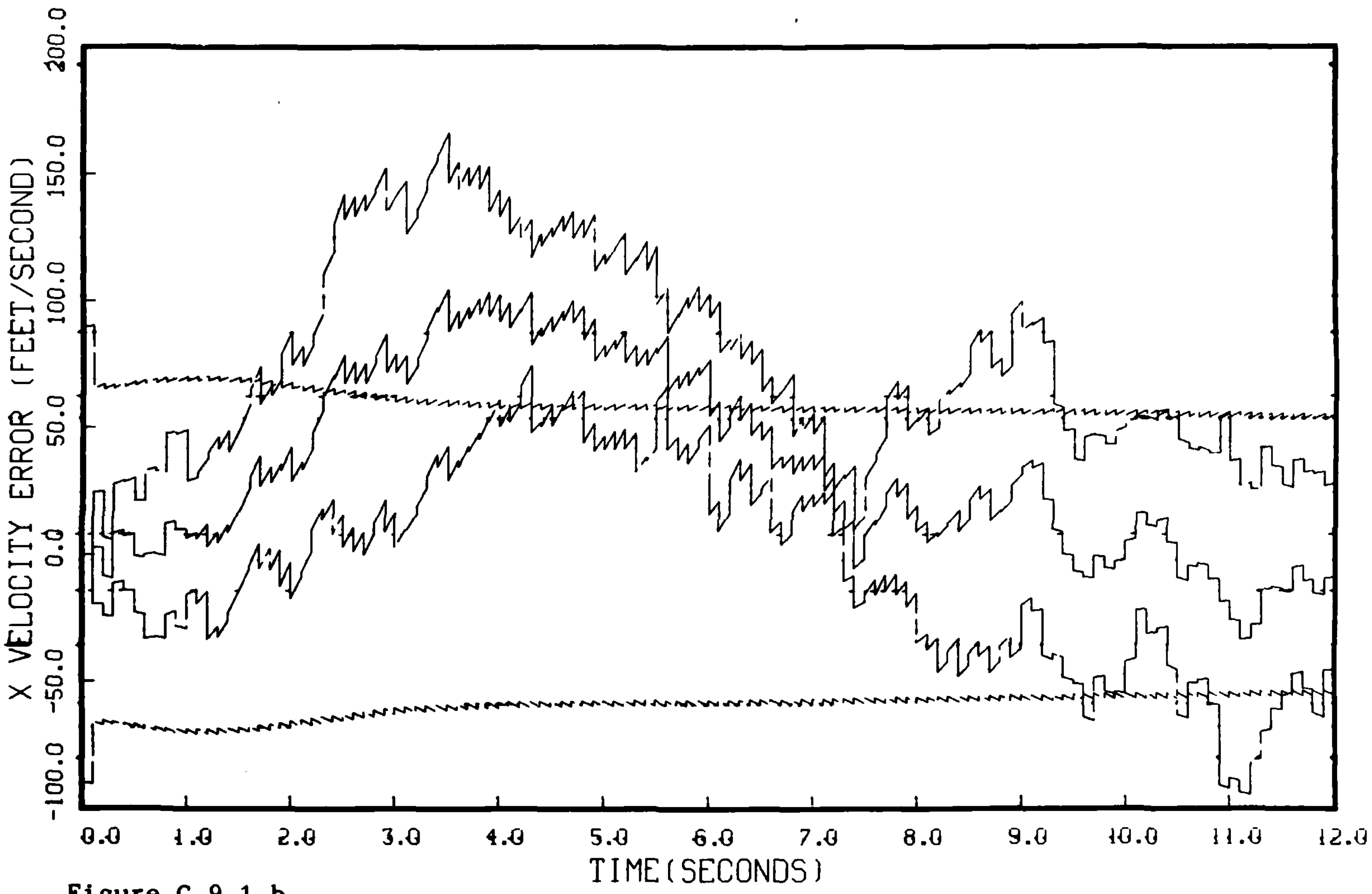


Figure G.9.1.b
STATE 2, O(1)-O(2)-O(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.1, 5 RUN, NO FIGHTER MANS

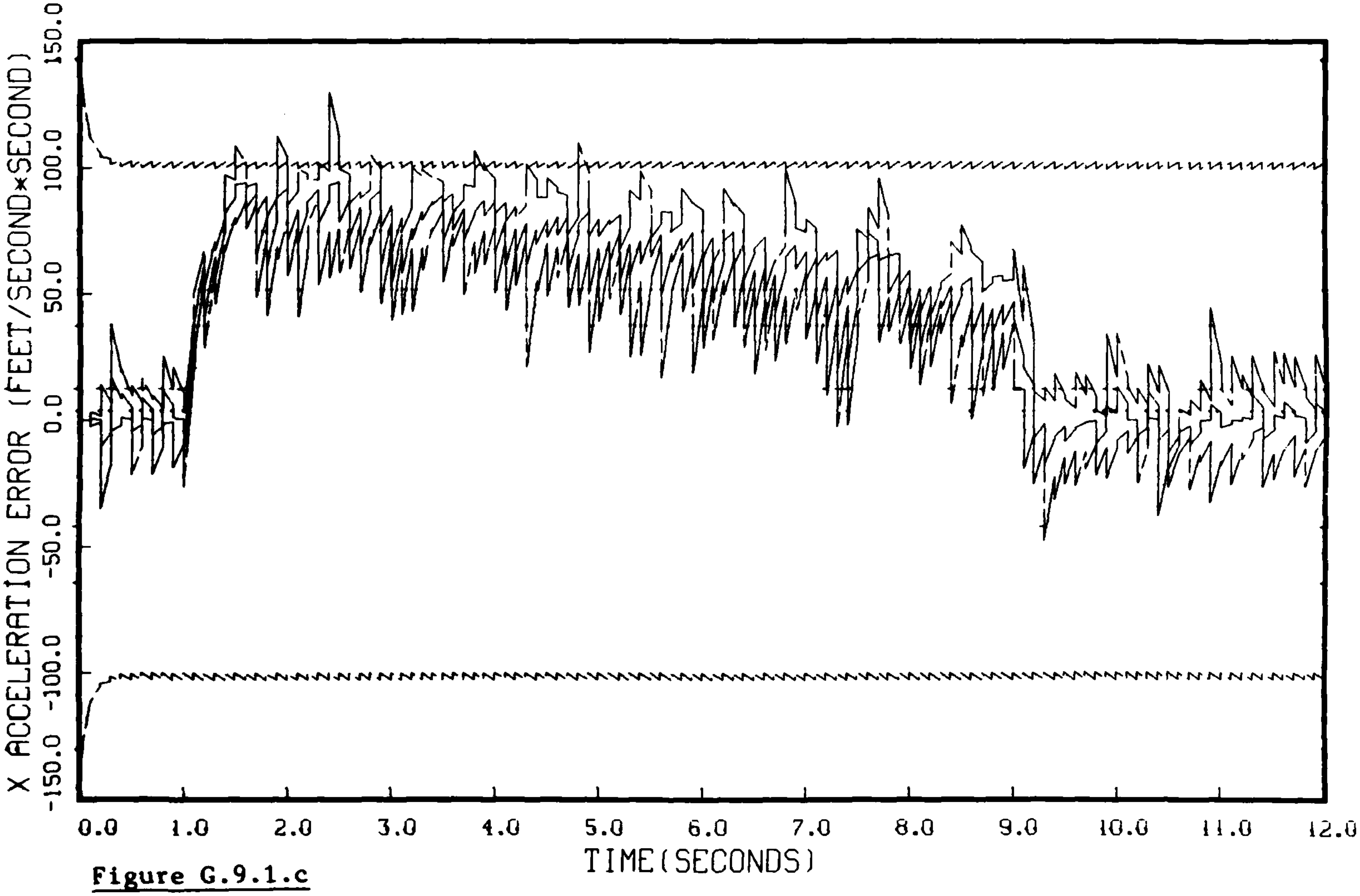


Figure G.9.1.c

STATE 3, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.1, 5 RUN, NO FIGHTER MANS

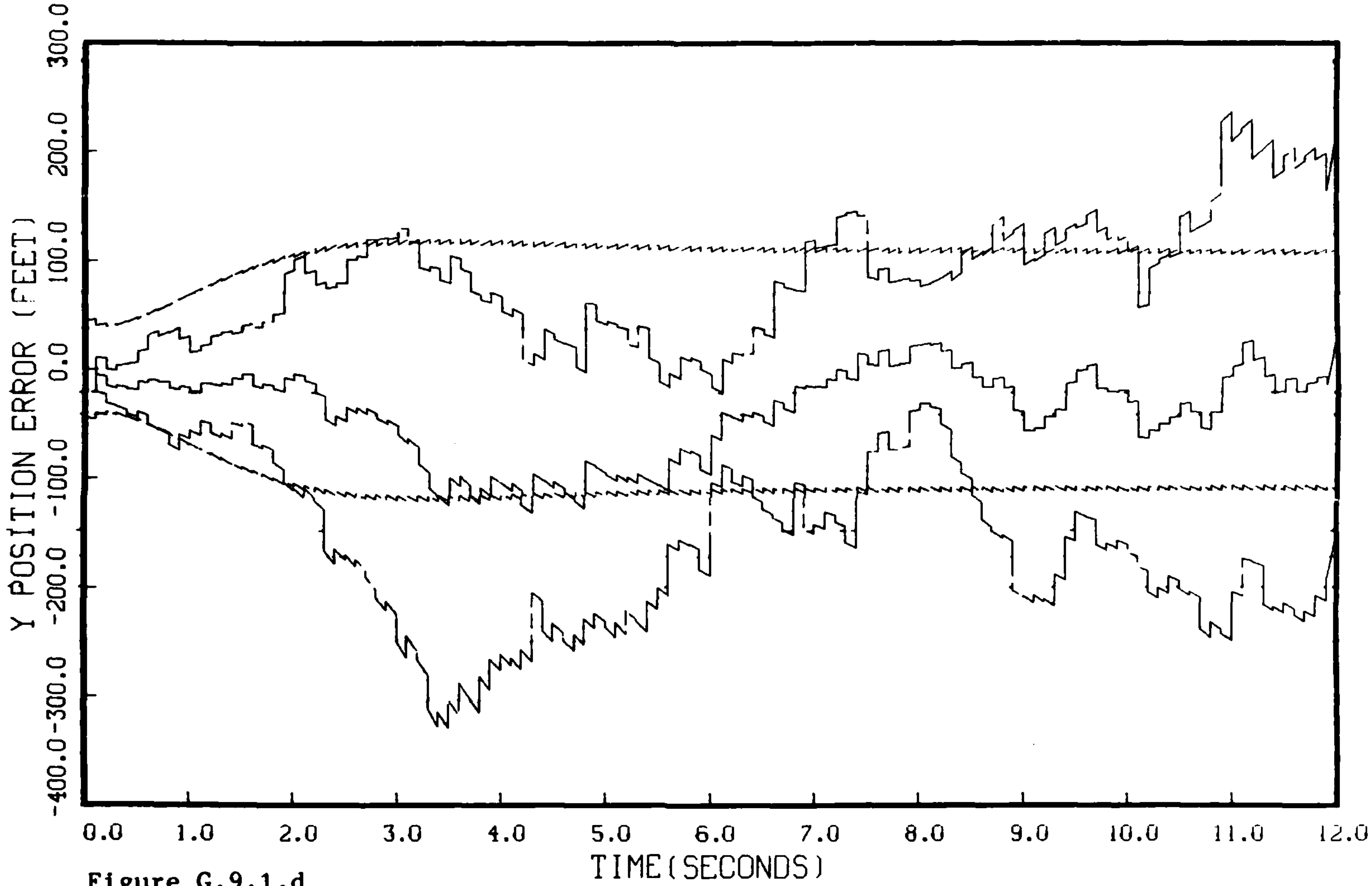


Figure G.9.1.d

STATE 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 5 RUN, NO FIGHTER MANS

SOFEPL DATE AND TIME - 85/10/18. 15.02.57. SOFE DATE AND TIME - 85/10/18. 14.04.03. WPAFB, DISSPLA, SOFEPL VERSION 2.?

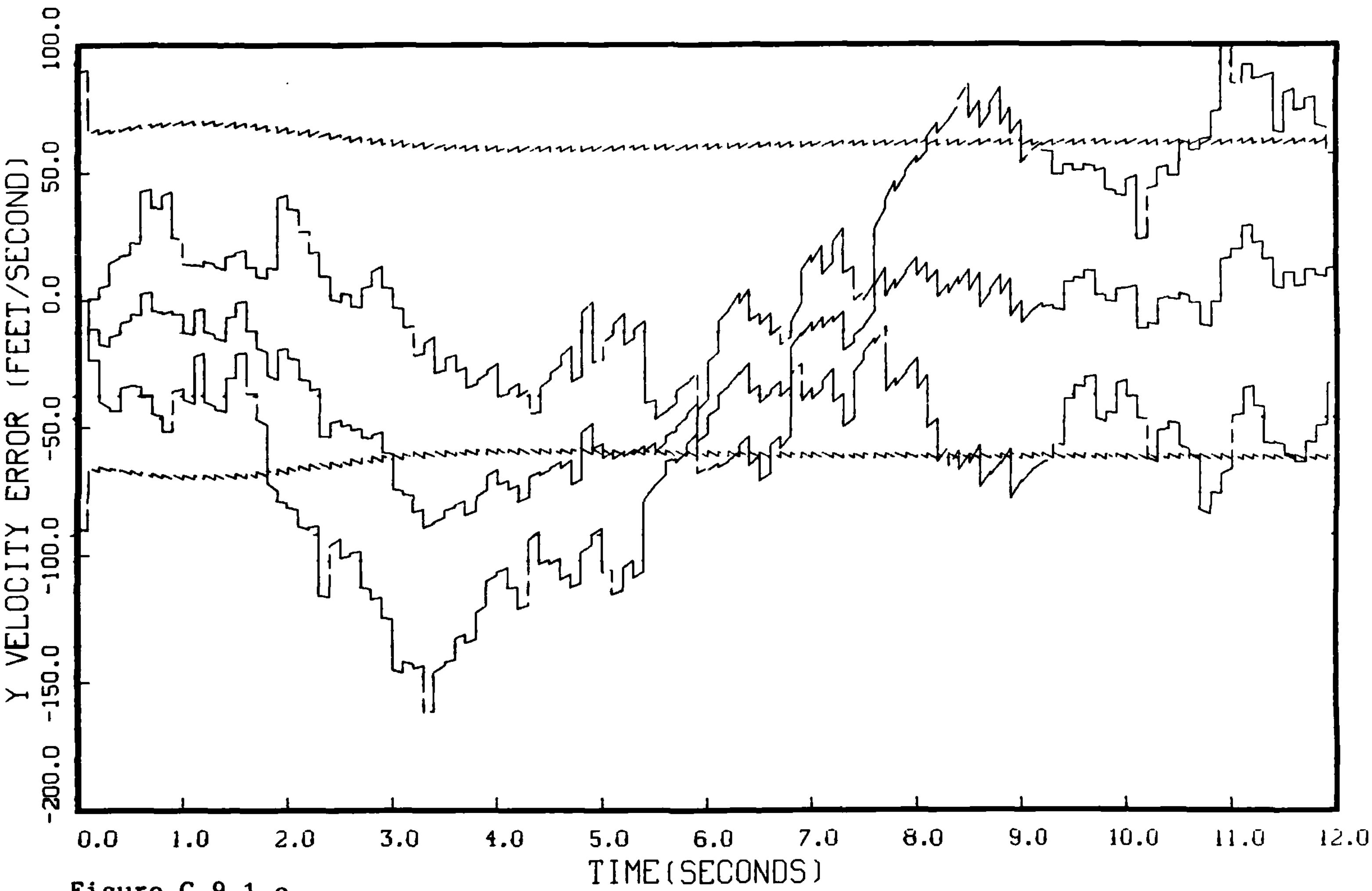


Figure G.9.1.e

STATE 5, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 5 RUN, NO FIGHTER MANS

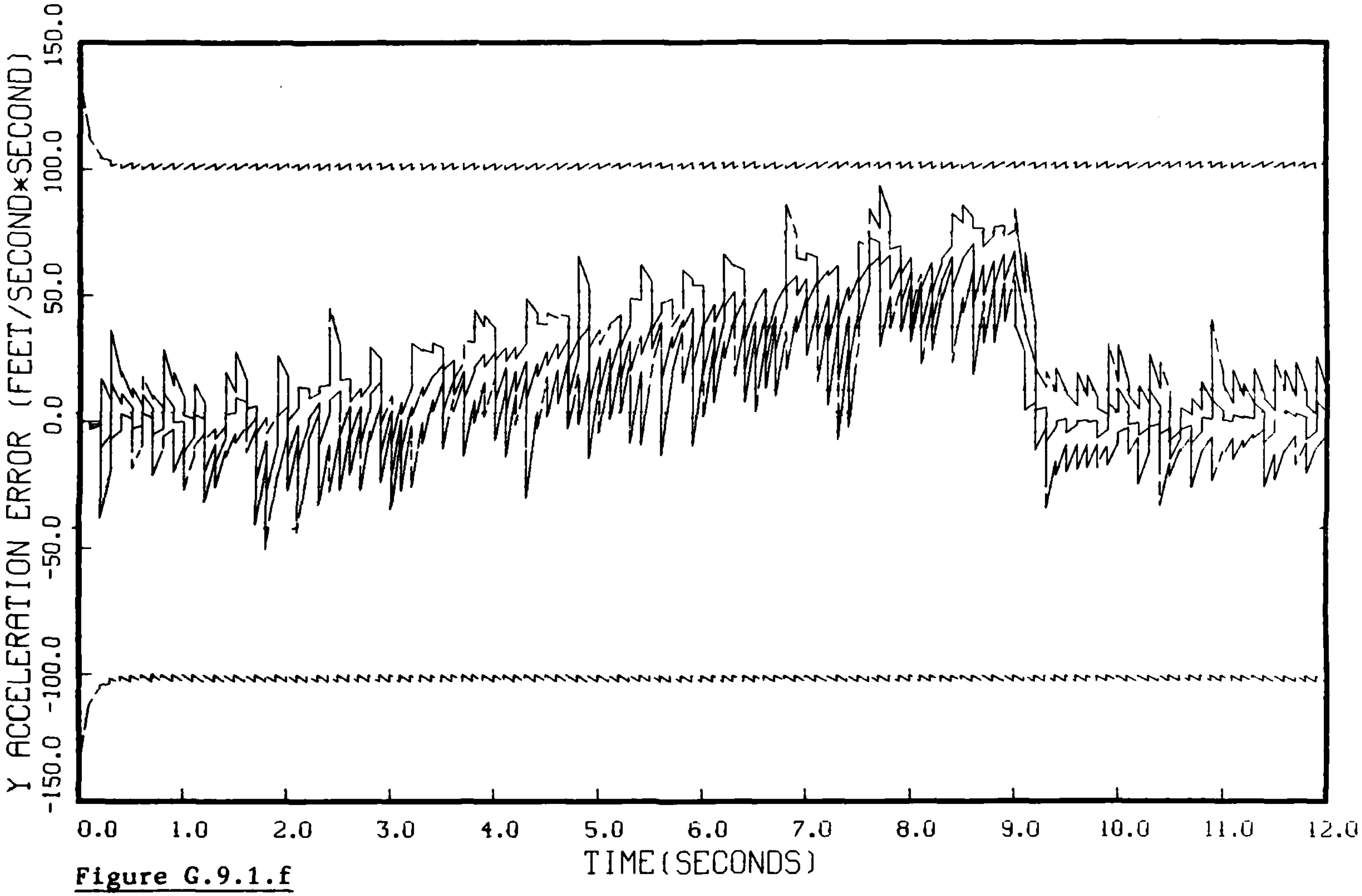


Figure G.9.1.f

STATE 6, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.1, 5 RUN, NO FIGHTER MANS

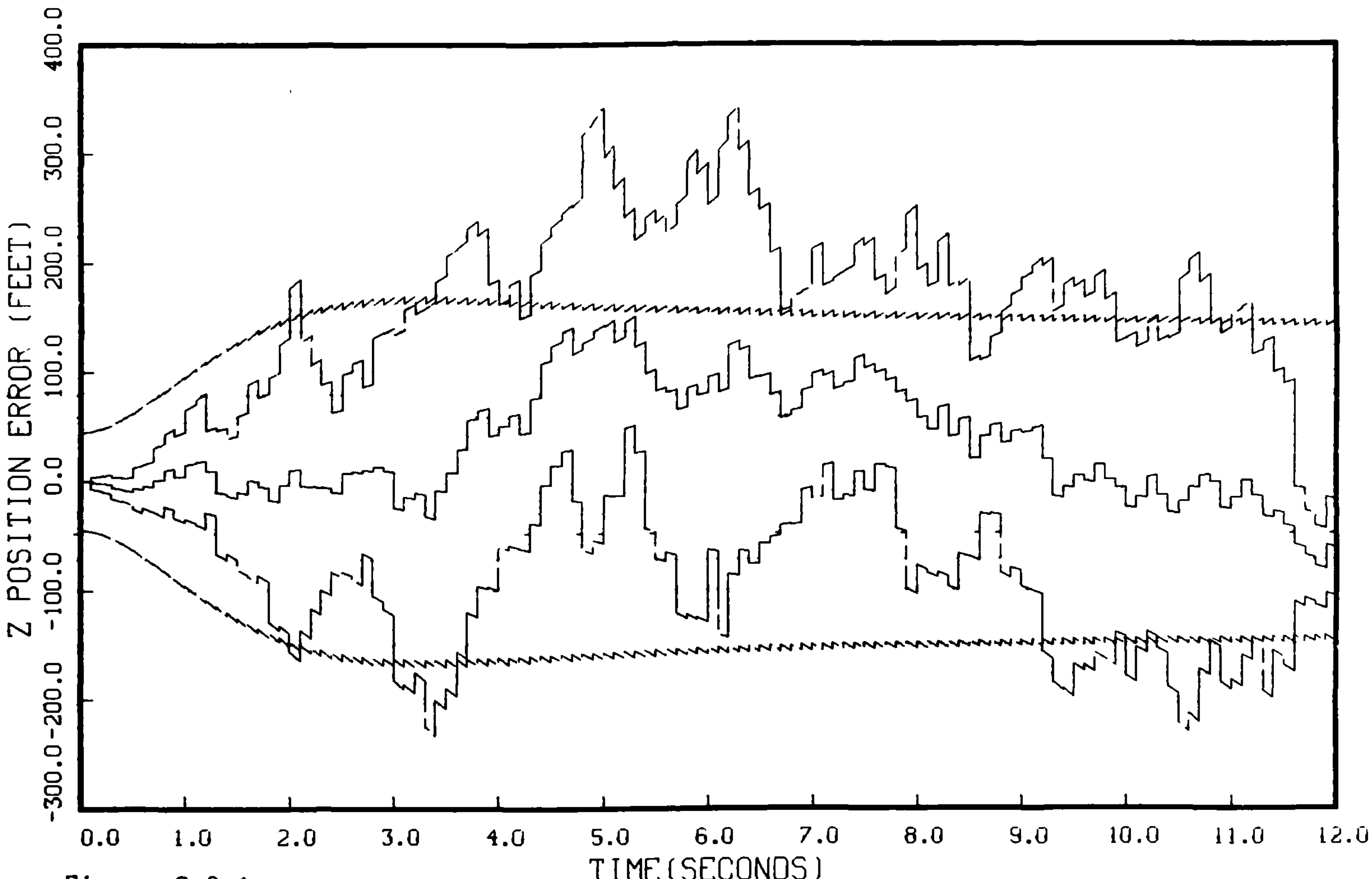


Figure G.9.1.g

STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.1, 5 RUN, NO FIGHTER MANS

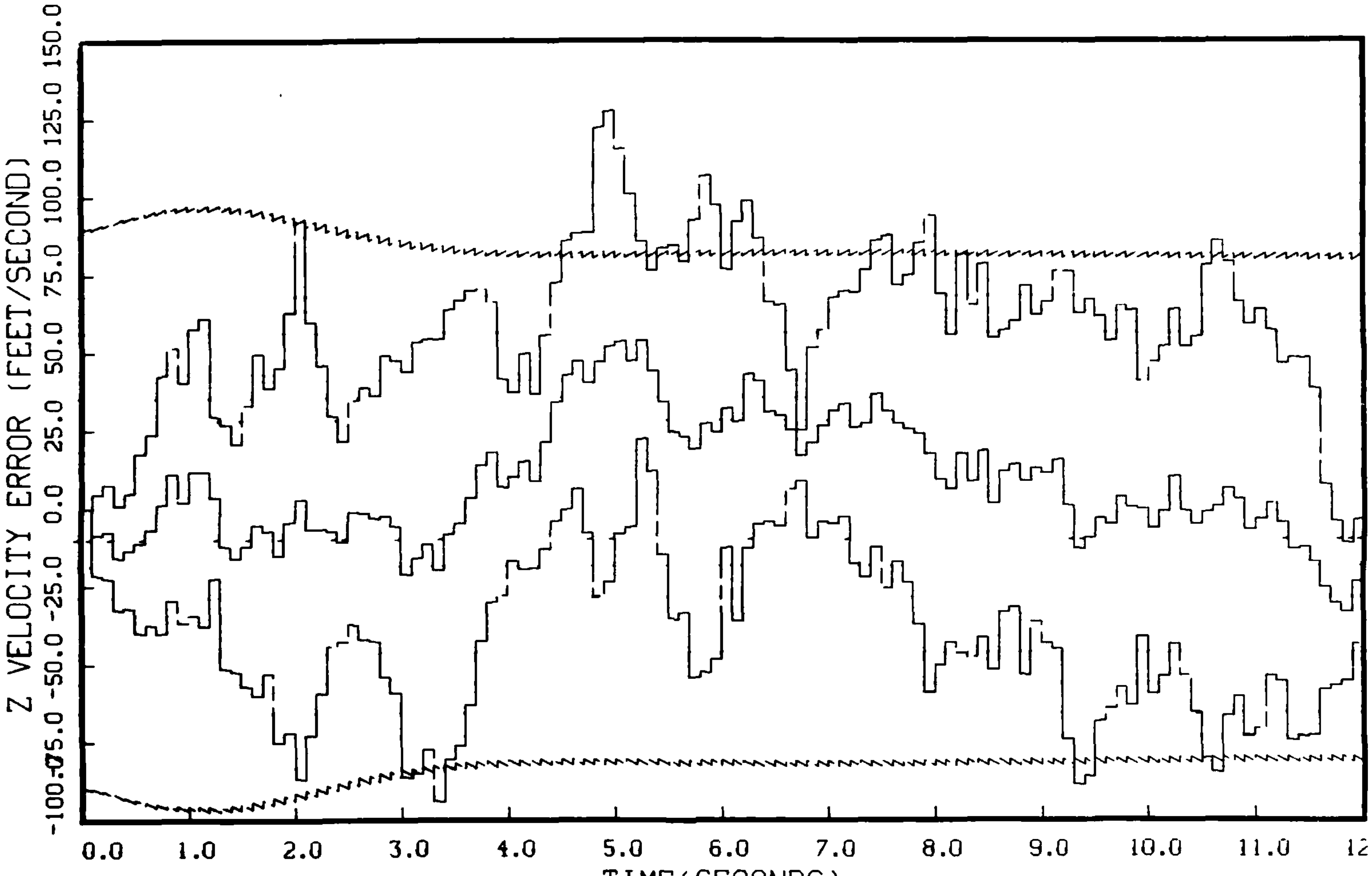


Figure G.9.1.h
STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 5 RUN, NO FIGHTER MANS

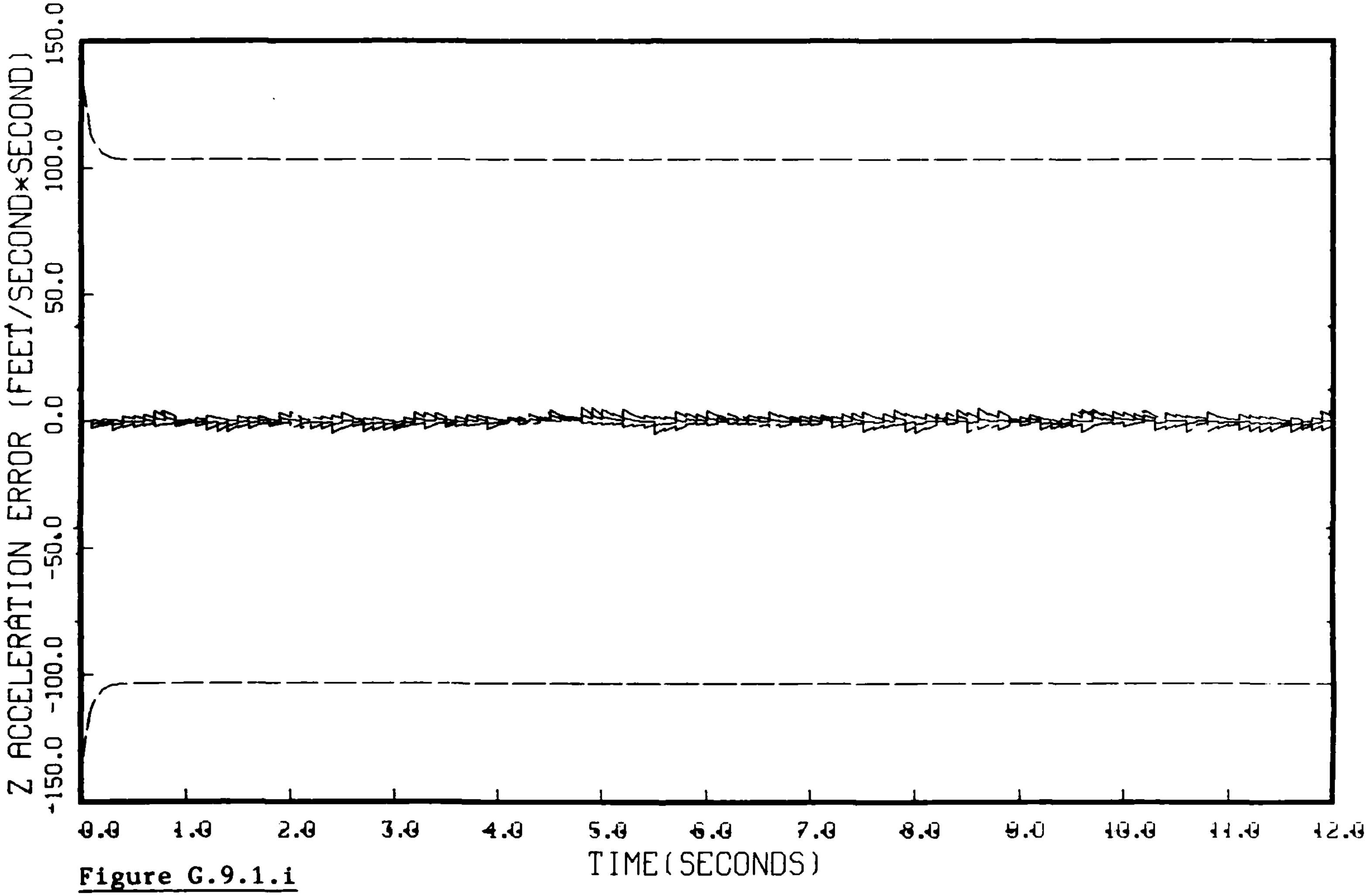


Figure G.9.1.i

STATE 9, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1, 5 RUN, NO FIGHTER MANS

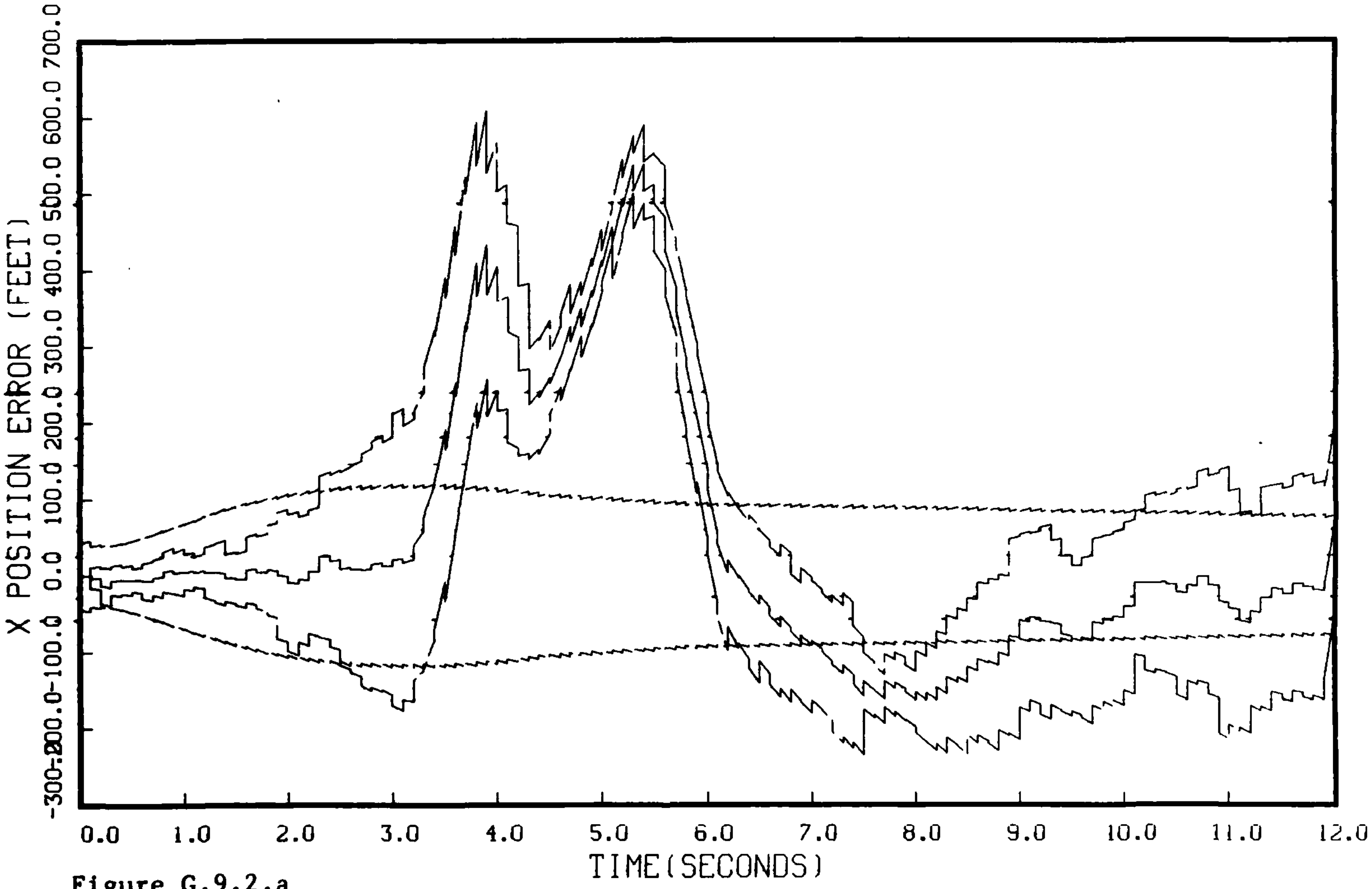


Figure G.9.2.a

STATE 1, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1,5 RUNS, NO TARGET MAN

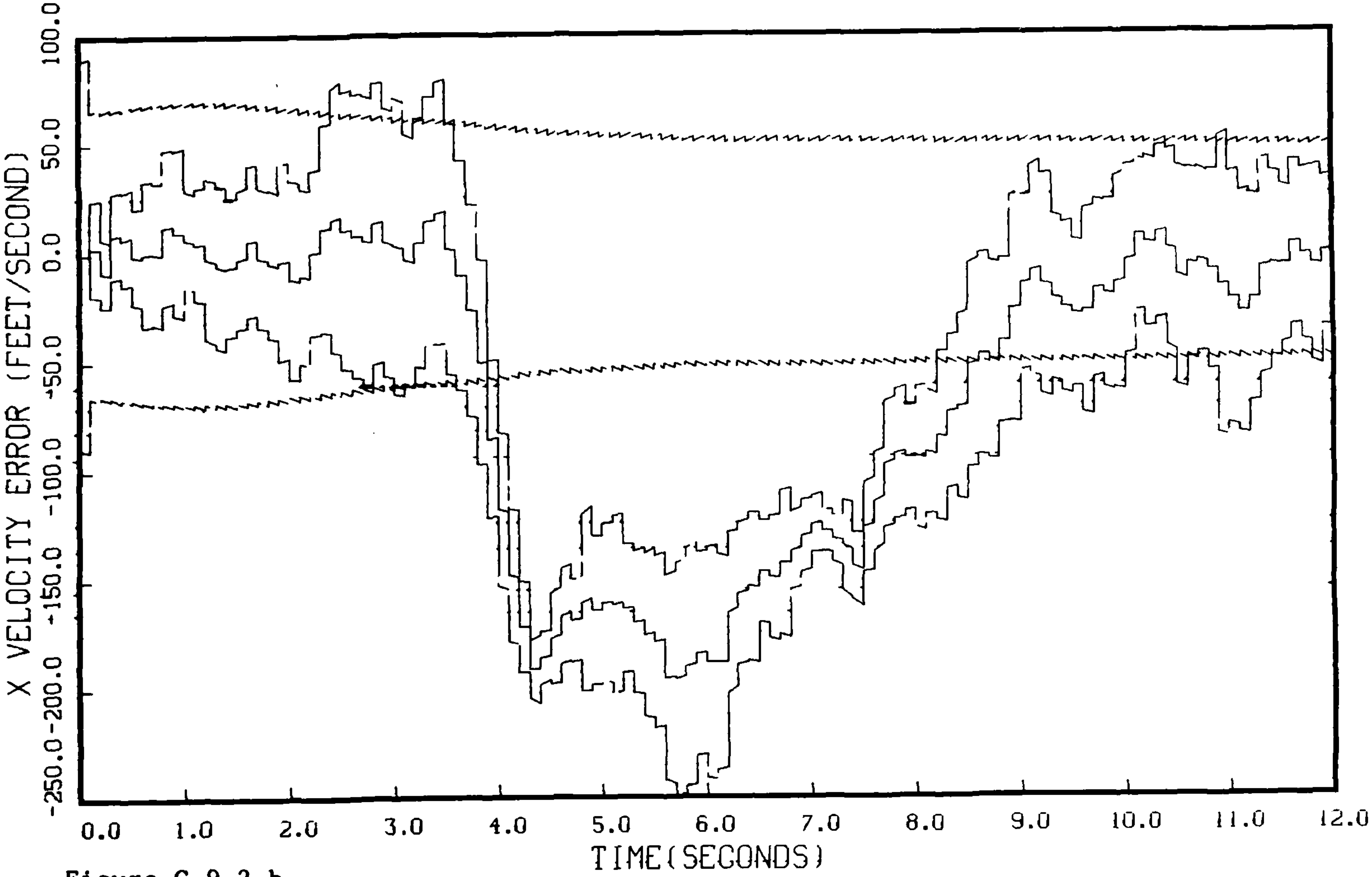


Figure G.9.2.b

STATE 2, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. ', UPDATE-0.1,5 RUNS, NO TARGET MAN

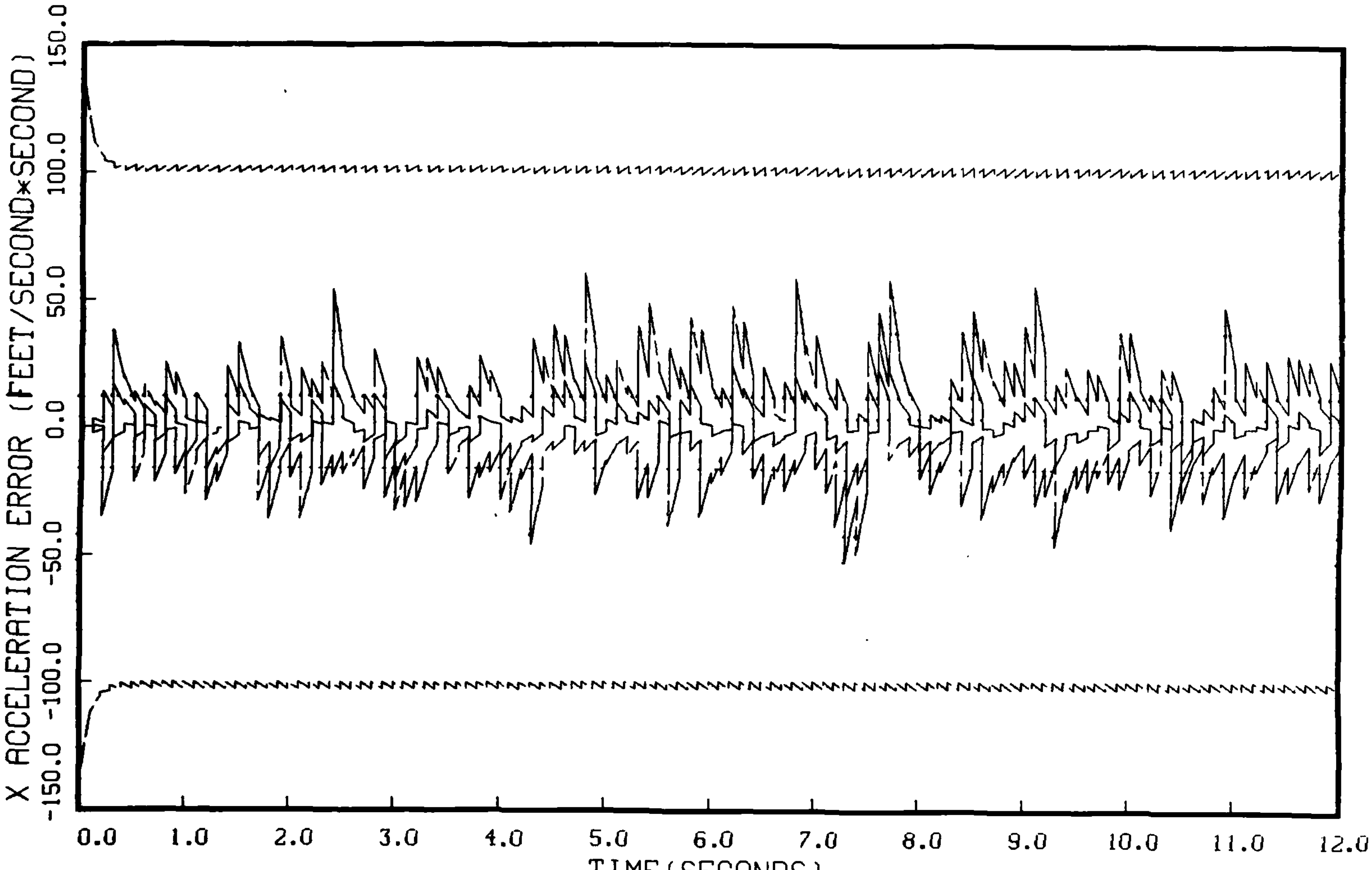


Figure G.9.2.c

STATE 3, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1,5 RUNS, NO TARGET MAN

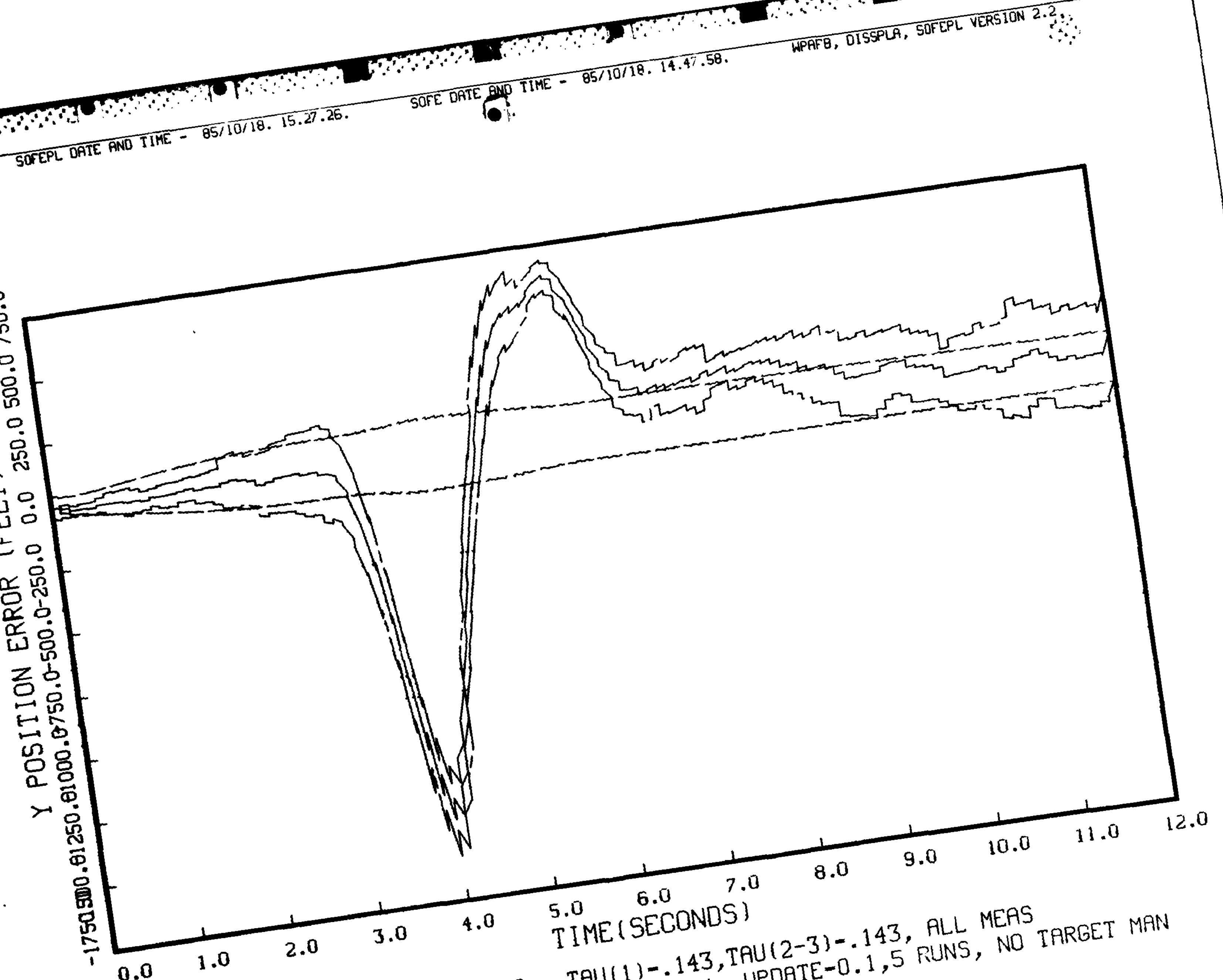


Figure G.9.2.d
STATE 4, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000. , UPDATE-0.1,5 RUNS, NO TARGET MAN

SOFEPL DATE AND TIME - 85/10/18. 15.27.26.   SOFE DATE AND TIME - 85/10/18. 14.47.58.   WPAFB, DISSPLA, SOFEPL VERSION 2.2

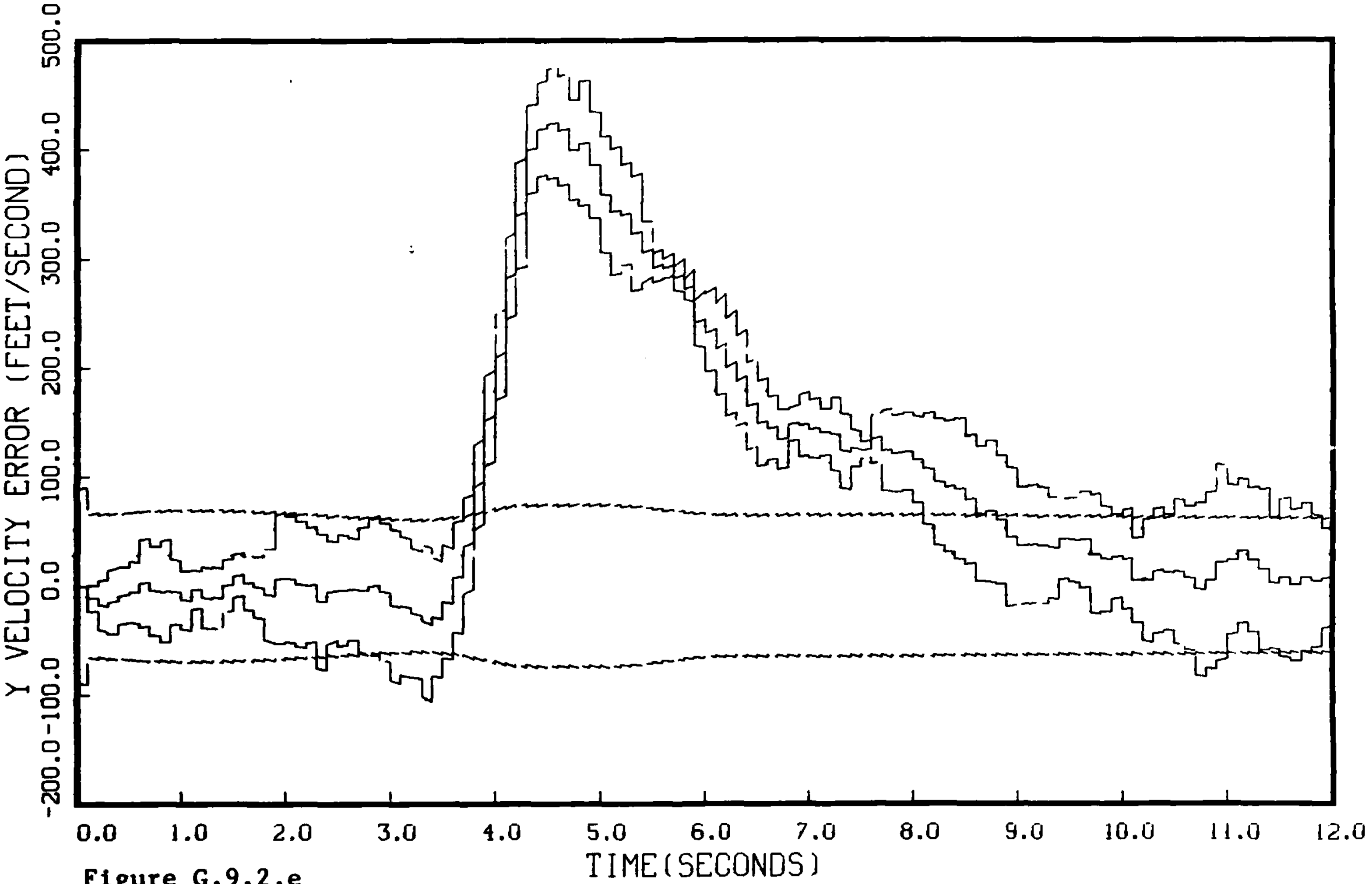


Figure G.9.2.e

STATE 5, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1,5 RUNS, NO TARGET MAN

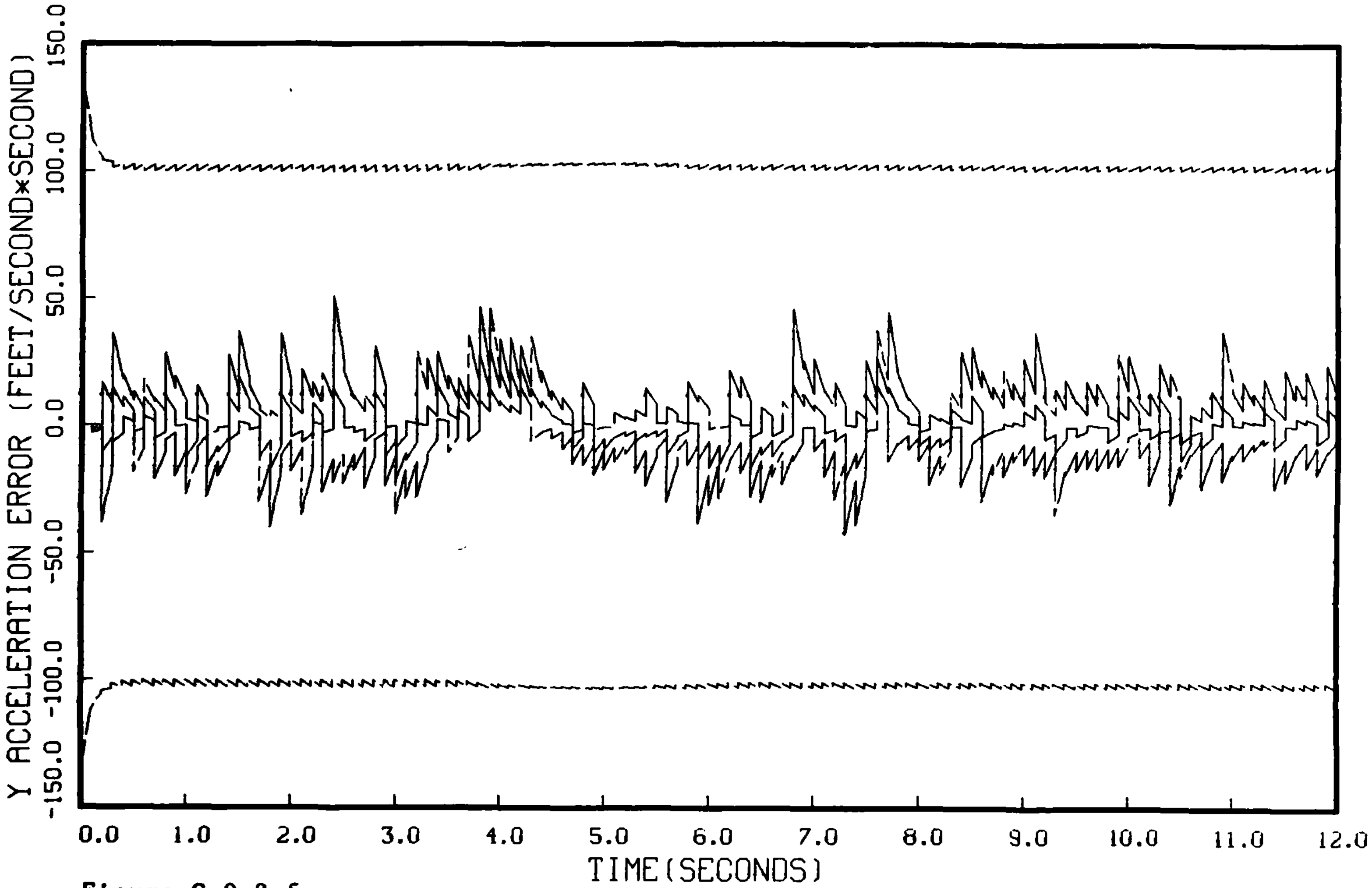


Figure G.9.2.f
STATE 6, O(1)-O(2)-O(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APO-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1,5 RUNS, NO TARGET MAN

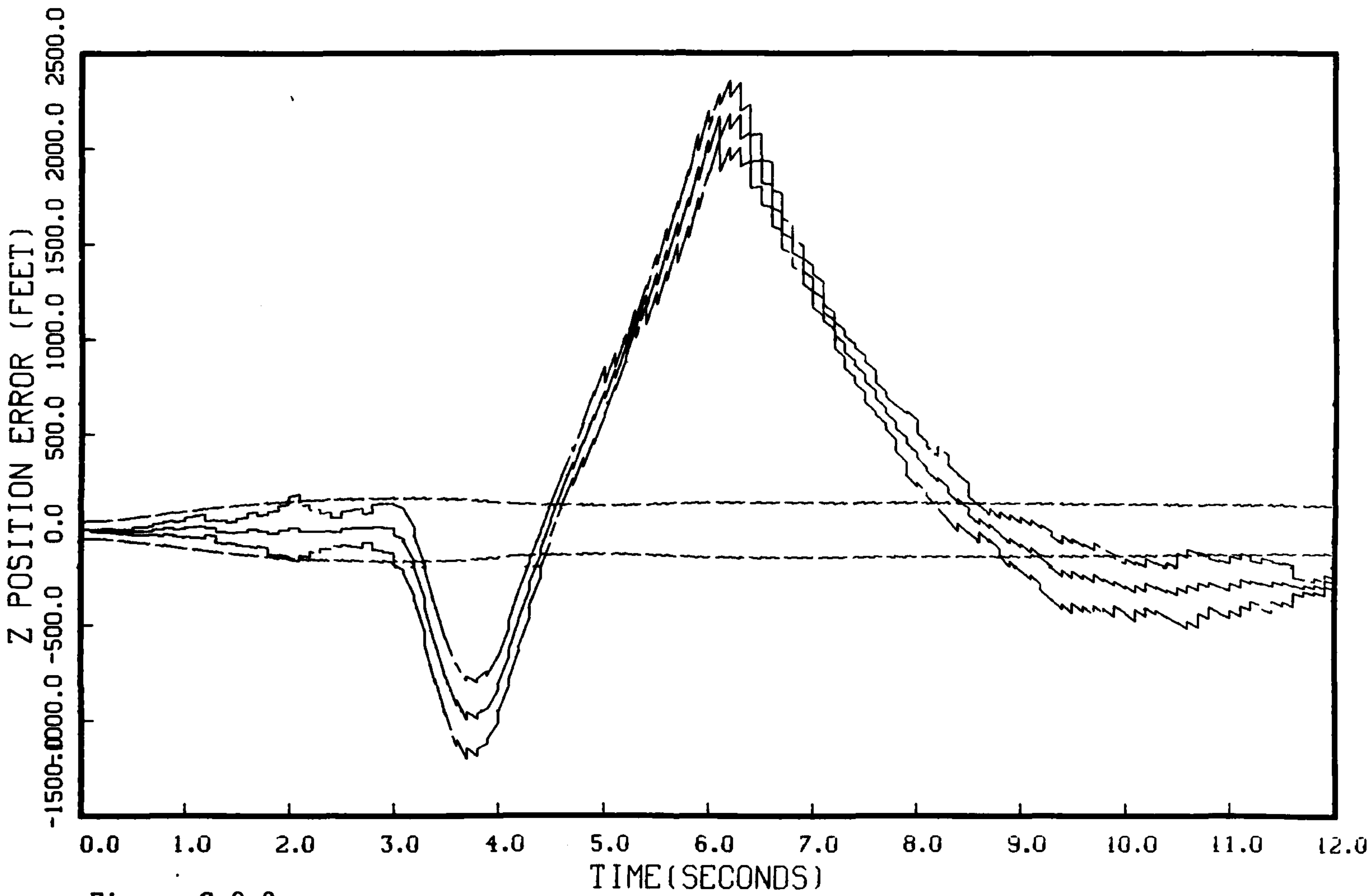


Figure G.9.2.g
STATE 7, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1,5 RUNS, NO TARGET MAN

8 SOFEPL DATE AND TIME - 85/10/18. 15.27.26. SOFE DATE AND TIME - 85/10/18. 14.47.58. WPAFB, DISSPLA, SOFEPL VERSION 2.2

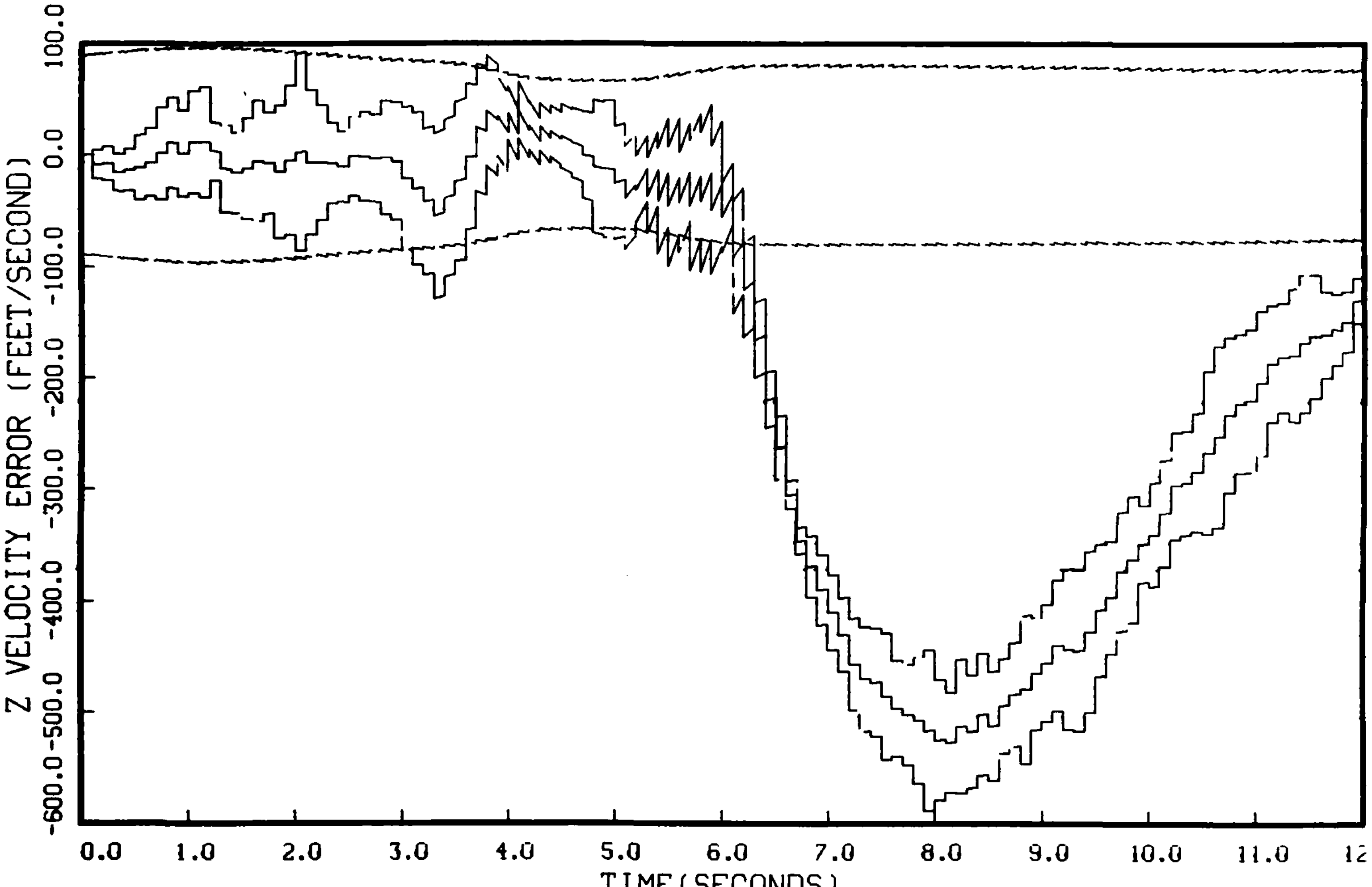


Figure G.9.2.h

STATE 8, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APG-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1,5 RUNS, NO TARGET MAN

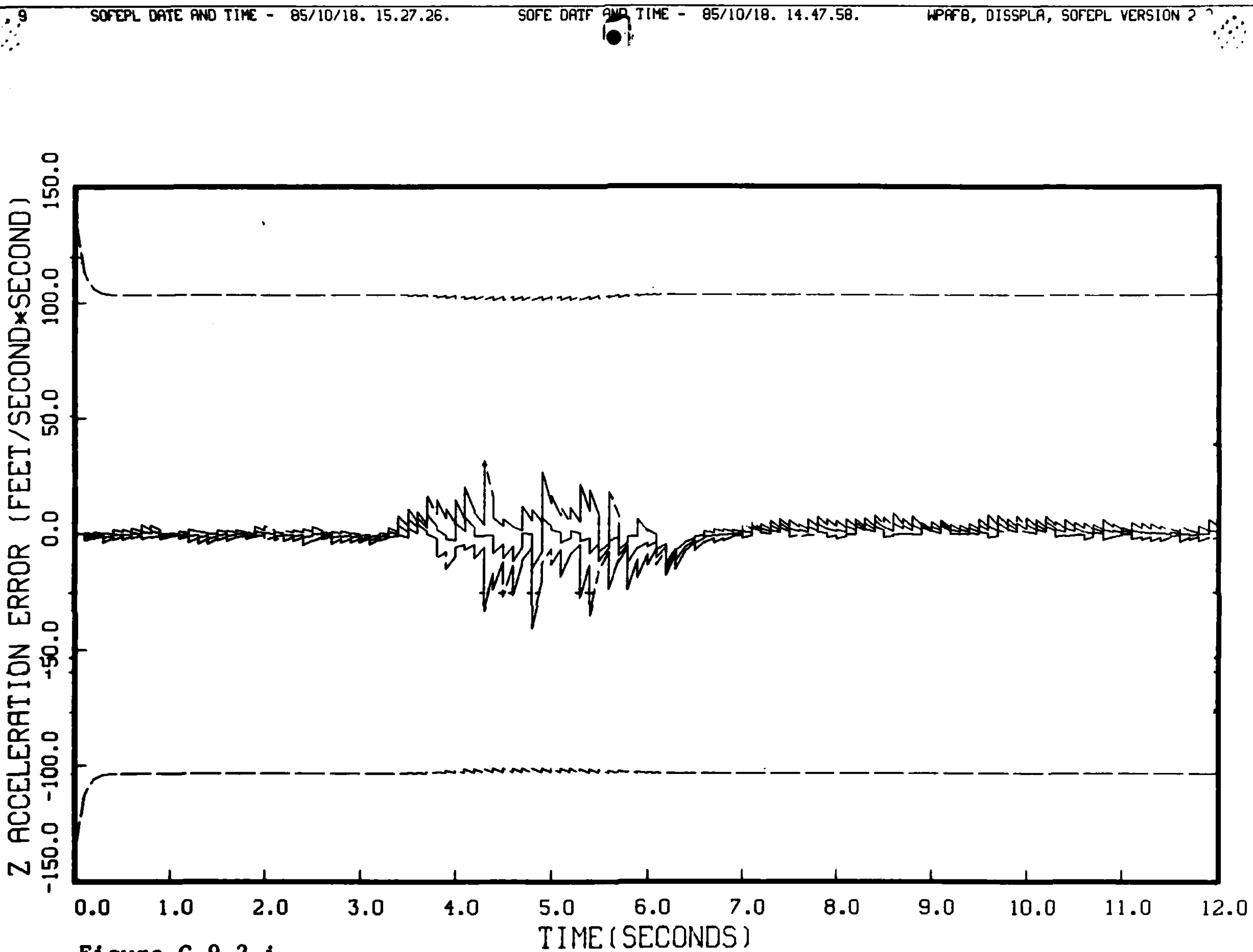


Figure G.9.2.i

STATE 9, Q(1)-Q(2)-Q(3)-149300., TAU(1)-.143,TAU(2-3)-.143, ALL MEAS
APQ-120, BEAM ATTACK, INITIAL RANGE-40,000.', UPDATE-0.1,5 RUNS, NO TARGET MAN

Figure Set G.9.3 is the same as Figure Set G.3.2

VITA

Captain Ross B. Anderson was born 29 April, 1952 in Los Angeles, California. He graduated from high school in Alta Loma, California in 1970 and from California State Polytechnic University, Pomona, California in 1979. Upon graduation, he received a commission in the USAF through Officer Training School. He has completed assignments as an Electrical Engineer at Headquarters TAC, Langley AFB, Virginia and as a Chief Engineer in the A-10 Program Management Office and the 314th Air Division, Osan AB, Korea, until entering the School of Engineering, Air Force Institute of Technology, in May 1984.

Permanent Address: 9129 Roberds Street
Alta Loma, CA 91701

SECURITY CLASSIFICATION OF THIS PAGE

AD-A1 [illegible] 234

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
Unclassified

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
AFIT/GE/ENG/85D-2
School of Engineering

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Engineering

6b. OFFICE SYMBOL *(If applicable)*
AFIT/ENG

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

7b. ADDRESS *(City, State and ZIP Code)*

8a. NAME OF FUNDING/SPONSORING ORGANIZATION
Air Force Logistics Cmd

8b. OFFICE SYMBOL *(If applicable)*
OO-ALC/MMECB

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
Unfunded

8c. ADDRESS *(City, State and ZIP Code)*
OO-ALC/MMECB
Hill AFB, Utah 84056

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.

11. TITLE *(Include Security Classification)*
See box 19

12. PERSONAL AUTHOR(S)
Ross B. Anderson, BSEEE, Capt, USAF

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ________

14. DATE OF REPORT *(Yr., Mo., Day)*
1985 December

15. PAGE COUNT
457

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.
09	03	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Extended Kalman Filter, F-4E/G Fire Control System, Target Estimation, State Transition Matrix, Monte Carlo Analysis

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: PRELIMINARY KALMAN FILTER DESIGN TO IMPROVE AIR COMBAT MANEUVERING TARGET ESTIMATION FOR THE F-4E/G FIRE CONTROL SYSTEM

Approved for public release: IAW AFR 190-17.
Lynn E. Wolaver 16 [illegible]
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AFSC)
Wright-Patterson AFB OH 45433

Thesis Chairman: Major William H. Worsley, Instructor of Electrical Engineering

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. □ DTIC USERS □

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL
Major William H. Worsley
Instructor of Electrical Engineering

22b. TELEPHONE NUMBER *(Include Area Code)*
(513) 255-2024

22c. OFFICE SYMBOL
AFIT/ENG

Currently, the F-4E/G uses a Wiener-Hopf filter for estimating target position, velocity, and acceleration during air combat maneuvering. As implemented, the errors between the actual target variables and the estimate of these variables are too large. The purpose of this study is to evaluate the feasibility of replacing the Wiener-Hopf filter with a Kalman filter in order to obtain better estimates. The evaluation is made by first designing an appropriate preliminary design Kalman filter and then testing the design through a Monte Carlo computer simulation analysis. The computer simulation results indicate that the Kalman filter is capable of significantly outperforming the Wiener-Hopf filter, and as such, should be developed into a final design.

The Kalman filter contains nine states (three relative target position, three total target velocity, and three total target acceleration states). Filter propagation is based on linear time-invariant dynamics primarily because of the limited capabilities of the on-board aircraft computer. The linear dynamics permits propagation by a state transition matrix. Measurement updates use six measurements (range, range rate, azimuth angle, elevation angle, azimuth rate, and elevation rate) available on the F-4. Both continuous time sampled-data and discrete-time sampled-data designs are included.

END

FILMED

3-86

DTIC